CICS Cobol Tutorial 3
Datenbankzugriff mit CICS (COBOL)

Ziel dieses Tutorials ist es, mittels einer CICS-Transaction auf die in Tutorial 4 erstellte DB2-Datenbank zuzugreifen. Unser Anwendungsprogramm soll wieder aus zwei Teilen bestehen, einem COBOL-Programm für die Business Logic und einem BMS-Programm für die Presentation Logic.

Hinweis: Tutorial 5 baut auf die erfolgreiche Bearbeitung von Tutorial 4 auf.

Unser Business Logic-Programm soll dabei SQL-Aufrufe enthalten. Diese müssen durch einen SQL-Precompiler in native DB2 API-Aufrufe übersetzt werden, ehe der COBOL-Compiler das Business Logic-Programm übersetzen kann.

1. Vorgehensweise

CICS trennt strikt Berechnungen und Datenbankzugriffe von dem Layout der Darstellungen auf Panels. Ersteres wird als "Business Logic" und letzteres als "Presentation Logic" bezeichnet.

Die Präsentationslogik wird , genauso wie in Tutorial 3, durch ein JCL Script erzeugt. Die Business Logik besteht aus einem Cobol Programm, welches auf die DB2 Datenbank zugreift.

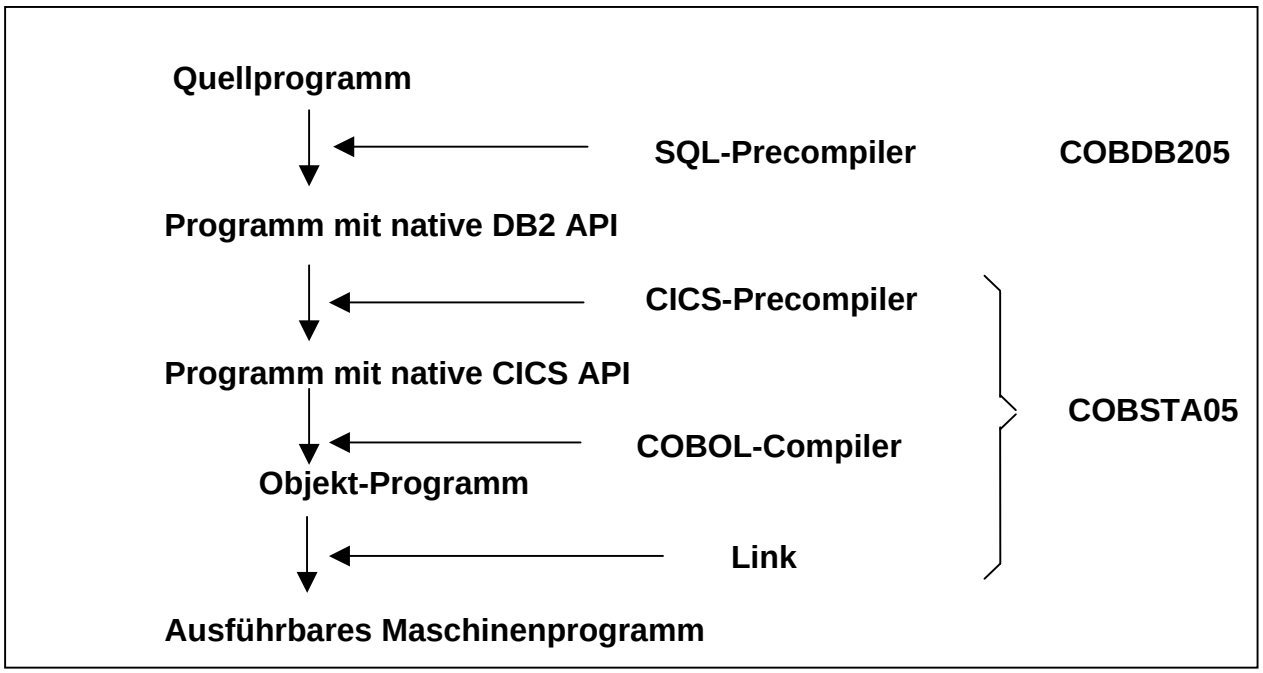

Abbildung 1:
Schritte vom COBOL-Programm mit EXEC SQL-Statements zum ausführbaren Maschinenprogramm

Wie in Abbildung 1 dargestellt, muss das fertige Cobol Quellprogramm vor der Übersetzung durch einen SQL Precompiler verarbeitet werden. Wir benutzen standardmäßig SQL für den Zugriff auf die DB2 Datenbank. Der SQL Precompiler übersetzt alle SQL Aufrufe durch die native API der DB2 Datenbank. Dieser native API Code wird ebenfalls in Cobol generiert. Das gesamte Cobol Programm wird dann wie in Tutorial 3 durch einen Compiler und einen Linkage Editor in ausführbaren Maschinencode übersetzt.

Unser Anwendungsprogramm besteht aus einem Cobol Programm und mehreren JCL Scripts. Wir bringen diese als Member in einem neuen partitioned Data Set PRAK218.CICSDB2.COB unter. Spezifisch benutzen wir diese Member:

Cobol CICS Business Logic Quellcode	**PRAK218.CICSDB2.COB(COB218)**
Cobol CICS Presentation Logic	**PRAK218.CICSDB2.COB(COBMAP5)**
JCL Script für DB2 Precompile Lauf	**PRAK218.CICSDB2.COB(COBDB205)**
JCL Script für Cobol Übersetzung	**PRAK218.CICSDB2.COB(COBSTA05)**

Der Dataset PRAK218.CICSDB2.COB muss neu angelegt werden.

In diesem wird als ersten Schritt in einem Member das auszuführende JCL-Script: PRAK218.CICSDB2.COB(COBMAP5) (s. die Abbildungen 4) angelegt, welches die Presentation Logic behandelt. Sie besteht aus genau einem Mapset "MSET218", der genau eine Map "MAP218" enthält. Ein Mapset kann aber auch mehrere Maps enthalten. Diese Map "MAP218" definiert Positionen, Länge sowie weitere Attribute der Darstellung der Daten aus der DB2-Datenbank auf dem Bildschirm.

Anschließend erstellen wir das COBOL-Programm "PRAK218.CICSDB2.COB(COB218)" (s. die Abbildung 6), welches EXEC SQL-Statements enthält.

So wie in Abbildung 1 dargestellt, führt das zweite von uns erstellte JCL-Script "PRAK218.CICSDB2.COB(COBDB205)" einen Precompiler Durchlauf aus. Alle EXEC SQL-Statements im COBOL-Programm werden durch dieses in native DB2 API-Aufrufe konvertiert.

Als drittes JCL-Script wird von uns "PRAK218.CICSDB2.COB(COBSTA05)" erstellt und ausgeführt (s. auch die Abbildung 12). Der CICS-Precompiler generiert aus dem COBOL-Programm mit native *DB2* API-Aufrufen ein COBOL-Programm mit native *CICS* API-Aufrufen. Anschließend wird der nun so entstandene COBOL-Programmcode zu einen Objekt-Programmcode übersetzt, aus welchem der Linker ein ausführbares Maschinenprogramm erzeugt (s. auch Abbildung 1).

Alle diese Schritte werden im TSO ausgeführt. TSO ist ein Subsystem von z/OS. Ein weiteres Subsystem von z/OS ist CICS. Der folgende Teil des Tutorials erläutert, über welche Schritte das übersetzte Cobol Programm innerhalb des CICS Subsystems als CICS-Transaktion mit der Transaktions-ID "X218" installiert wird. Folgende Schritte sind dazu notwendig:

1. **Definition des Mapsets mittels "CEDA DEFINE MAPSET(MSET218) GROUP(PRAK218)"**
2. **Definition des COBOL-Programmes mittels "CEDA DEFINE PROG(COB218) GROUP(PRAK218)"**
3. **Definition des Namens der Transaction-ID mittels "CEDA DEFINE TRANS(X218) GROUP(PRAK218)"**
4. **Definition unserer Datenbank und Datenbanktabelle mittels "CEDA DEFINE DB2ENTRY"**

Nach diesen Schritten sind der Mapset MSET218 mit der Map MAP218, das ausführbare Maschinenprogramm, das aus COB218 generiert wurde, die selbst definierte Transaction-ID "X218" sowie die Datenbank und Tabelle, aus der ausgelesen werden soll, dem CICS-System bekannt. Ebenfalls ist ihm bekannt, dass alle diese Komponenten der Gruppe "PRAK218" zugewiesen wurden. Diese Gruppe wird durch Schritt 1. automatisch erstellt. Doch diese *Definitionen* reichen noch nicht aus. Unser Ziel erreichen wir erst, wenn alle Komponenten auch *installiert* werden. Dies geschieht durch

5. **"CEDA INSTALL GROUP(PRAK218)"**

Nun haben wir unser Ziel erreicht. Geben wir "X218" unter CICS ein (s.

Abbildung 30), so wird unsere selbst definierte Transaktion ausgeführt, welche die Spalten "VORNAME" und "NACHNAME" aus der im Tutorial 4 angelegten DB2-Tabelle ausliest und auf unserem Bildschirm ausgibt (s.
Abbildung 31).
Warnung:

Ihr DB2ENTRY ist nur ein einziges Mal von Ihnen installierbar. Deshalb könnte z.B. Ihre zweite Anwendung von "CEDA INSTALL GROUP ..." die Fehlermeldung "INSTALL UNSUCCESSFUL" produzieren. Dieser Fehler kann von Ihnen mangels Ihrer CICS-Zugriffsrechte nicht behoben werden. Im Anhang wird dieses Problem erläutert. Informieren Sie deshalb umgehend Ihren Betreuer.

2. Anlegen des Mapsets

Die nun benötigten Datasets "PRAK218.DBRMLIB.DATA", "PRAK218.LIB" und "PRAK218.CICSDB2.COB" müssen neu angelegt werden. "PRAK218.DBRMLIB.DATA" wird während der Ausführung des SQL-Precompilers automatisch gefüllt. "PRAK218.CICSDB2.COB" nimmt die von uns zu erstellenden Quellprogramme auf.

Aufgabe: *Legen Sie den Dataset "PRAK218.CICSDB2.COB" und - wenn noch nicht vorhanden - die Datasets "PRAK218.LIB" sowie "PRAK218.DBRMLIB.DATA" an. Verwenden Sie dazu folgende Parameter:*

Space units . . . . . KILOBYTE	*Record format . . . . FB*
Primary quantity . . 16	*Record length 80*
Secondary quantity 1	*Block size 320*
Directory blocks . . 2	*Data set name type : PDS*

Falls einer der beiden Datasets "PRAK218.LIB" oder "PRAK218.DBRMLIB.DATA" auf ihrem Account schon vorhanden ist, empfiehlt sich das Löschen aller Members sowie ein anschließender Compress dieser Datasets.

Mit Hilfe des "Edit Entry Panels" erstellen wir einen neuen Member "COBMAP5" (s. Abbildung 3) und bestätigen anschließend mit der Eingabetaste.

```
   Menu  RefList  RefMode  Utilities  Workstation  Help
--------------------------------------------------------------------------------
                              Edit Entry Panel

ISPF Library:
   Project . . . PRAK218
   Group . . . . CICSDB2  . . .          . . .          . . .
   Type  . . . . COB
   Member  . . . COBMAP5         (Blank or pattern for member selection list)

Other Partitioned, Sequential or VSAM Data Set:
   Data Set Name . . .
   Volume Serial . . .           (If not cataloged)

Workstation File:
   File Name . . . . .
                                       Options
Initial Macro  . . . .                    Confirm Cancel/Move/Replace
Profile Name . . . . .                    Mixed Mode
Format Name  . . . . .                    Edit on Workstation
Data Set Password  . .                    Preserve VB record length
```

Abbildung 2: Anlegen des Members "COBMAP5"

Unsere CICS-Anwendung soll wiederum aus einem BMS-Programm (Mapset) für die "Presentation Logic" und einem COBOL-Programm für die Business Logic bestehen. Wir beginnen mit dem Mapset.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAK218.CICSDB2.COB(COBMAP5) - 01.06            Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100 //PRAK218M JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000200 //         REGION=4M
000300 //ASSEM EXEC DFHMAPS,MAPNAME='MSET218',RMODE=24
000400 //COPY.SYSUT1  DD  *
000500 MSET085  DFHMSD TYPE=MAP,MODE=INOUT,LANG=COBOL2,STORAGE=AUTO,          *
000510               TIOAPFX=YES
000600 *    MENU  MAP
000700 MAP218   DFHMDI SIZE=(24,80),CTRL=(PRINT,FREEKB)
001000          DFHMDF POS=(9,13),ATTRB=(ASKIP,NORM),LENGTH=20,              *
001100               INITIAL='VORNAME             '
001200          DFHMDF POS=(9,34),ATTRB=(ASKIP,NORM),LENGTH=20,              *
001210               INITIAL='NACHNAME'
001300 VNAM1    DFHMDF POS=(11,13),ATTRB=(ASKIP,NORM),LENGTH=20
001400 NNAM1    DFHMDF POS=(11,34),ATTRB=(ASKIP,NORM),LENGTH=20
001500 VNAM2    DFHMDF POS=(12,13),ATTRB=(ASKIP,NORM),LENGTH=20
001600 NNAM2    DFHMDF POS=(12,34),ATTRB=(ASKIP,NORM),LENGTH=20
001700 VNAM3    DFHMDF POS=(13,13),ATTRB=(ASKIP,NORM),LENGTH=20
001800 NNAM3    DFHMDF POS=(13,34),ATTRB=(ASKIP,NORM),LENGTH=20
001900 VNAM4    DFHMDF POS=(14,13),ATTRB=(ASKIP,NORM),LENGTH=20
002000 NNAM4    DFHMDF POS=(14,34),ATTRB=(ASKIP,NORM),LENGTH=20
002100          DFHMSD TYPE=FINAL
002200          END
002300 /*
002400 //
****** **************************** Bottom of Data ****************************
Command ===>                                                Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
===============================================================================
```

Abbildung 3: Das BMS-Programm

Dies ist das vollständige BMS-Programm nach Fertigstellung.

Die Zeilen 700 bis 1210 definieren eine Überschrift, die aus 2 Feldern besteht. Die beiden Felder werden mit den Werten VORNAME und NACHNAME initialisiert.

Die Zeilen 1300 bis 2000 definieren 8 Felder, welche die Vornamen und Nachnamen von 4 Personen aufnehmen sollen, welche wir aus unserer DB2-Datenbank auslesen.

Wir geben "SUB" auf der Kommandozeile ein. Zusätzlich zu dem übersetzten Programm wird in dem Member "PRAK218.LIB(MSET218)" ein Template für unser Business Logic-Programm abgespeichert.

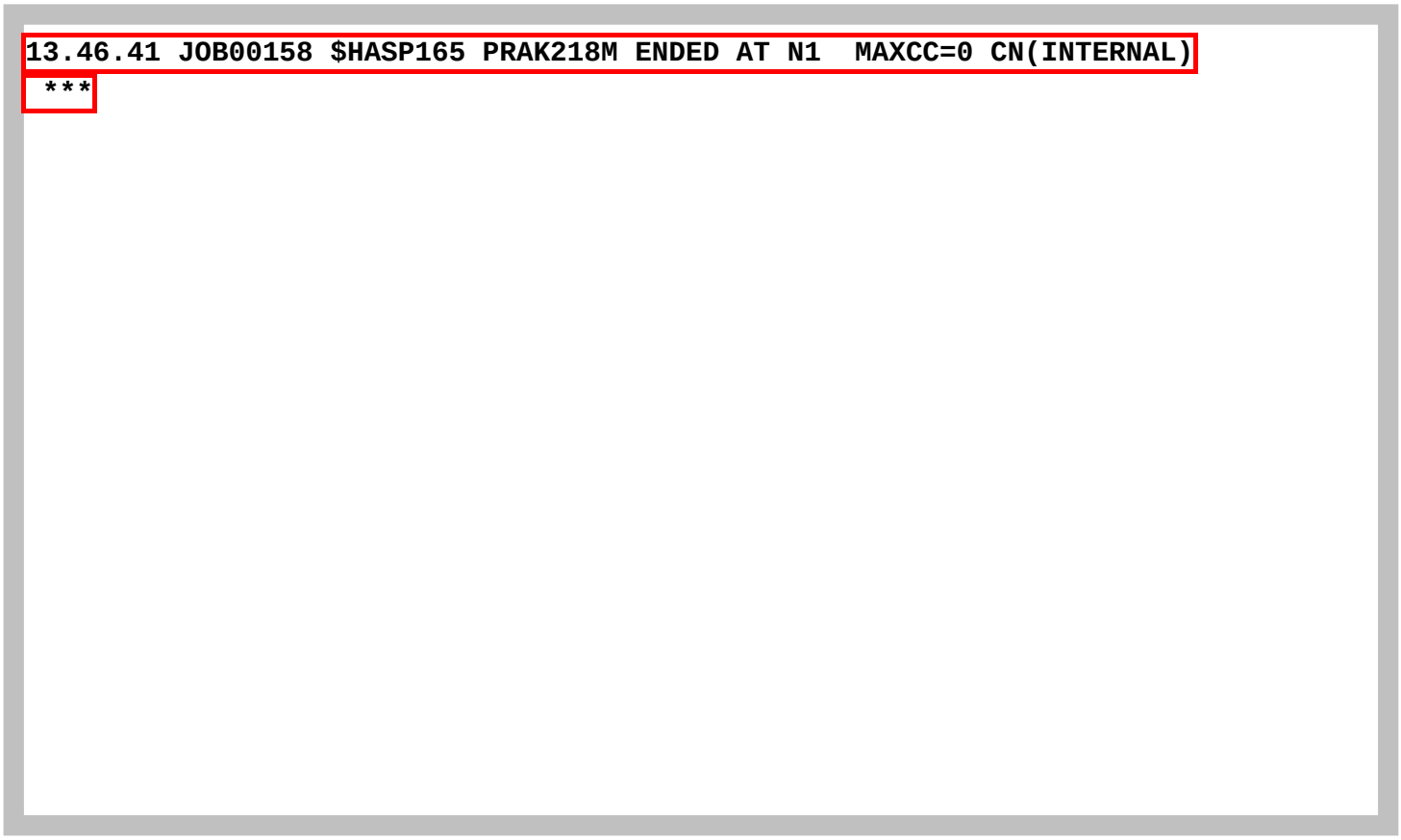

Abbildung 4: Bestätigung der Jobverarbeitung

Wir warten, bis JES unser BMS-Programm übersetzt hat (30-60 Sekunden). Durch das Betätigen der Eingabetaste erscheint der hier gezeigte Panel (s.Abbildung 7). "MAXCC=0" bestätigt, dass die Übersetzung erfolgreich war.

Die Eingabetaste bringt uns zurück zum vorhergehenden Screen.

Aufgabe: ***Legen Sie einen Member an, schreiben Sie das BMS-Programm und führen Sie es aus. Ersetzen Sie "//PRAK218M" entsprechend Ihres Mainframe-Accountnamens. Benutzen Sie MAP<Ihre Prakt-ID> als Mapnamen sowie SET<Ihre Prakt-ID> als Mapsetnamen. Haben Sie z.B. den Account PRAK162, so ist Ihr Map-Name MAP162 und Ihr Mapset-Name MSET162.***

Wir betätigen zweimal die F3-Taste, um diesen Bildschirm zu verlassen.

Als nächstes sehen wir uns die Members von "PRAK218.LIB" an.

Wir wechseln zu dem Partitioned Dataset "PRAK218.LIB". Dort existiert jetzt der während der Übersetzung erstellte Member "PRAK218.LIB(MSET5218)". Wir schauen uns "PRAK218.LIB(MSET5218)" an.

```
000001        01  MAP218I.
000002            02  FILLER PIC X(12).
000003            02  VNAM1L    COMP  PIC  S9(4).
000004            02  VNAM1F    PICTURE X.
000005            02  FILLER REDEFINES VNAM1F.
000006              03 VNAM1A    PICTURE X.
000007            02  VNAM1I  PIC X(20).
000008            02  NNAM1L    COMP  PIC  S9(4).
000009            02  NNAM1F    PICTURE X.
000010            02  FILLER REDEFINES NNAM1F.
000011              03 NNAM1A    PICTURE X.
000012            02  NNAM1I  PIC X(20).
000013            02  VNAM2L    COMP  PIC  S9(4).
000014            02  VNAM2F    PICTURE X.
000015            02  FILLER REDEFINES VNAM2F.
000016              03 VNAM2A    PICTURE X.
000017            02  VNAM2I  PIC X(20).
000018            02  NNAM2L    COMP  PIC  S9(4).
000019            02  NNAM2F    PICTURE X.
000020            02  FILLER REDEFINES NNAM2F.
000021              03 NNAM2A    PICTURE X.
000022            02  NNAM2I  PIC X(20).
000023            02  VNAM3L    COMP  PIC  S9(4).
000024            02  VNAM3F    PICTURE X.
000025            02  FILLER REDEFINES VNAM3F.
000026              03 VNAM3A    PICTURE X.
000027            02  VNAM3I  PIC X(20).
000028            02  NNAM3L    COMP  PIC  S9(4).
000029            02  NNAM3F    PICTURE X.
000030            02  FILLER REDEFINES NNAM3F.
000031              03 NNAM3A    PICTURE X.
000032            02  NNAM3I  PIC X(20).
000033            02  VNAM4L    COMP  PIC  S9(4).
000034            02  VNAM4F    PICTURE X.
000035            02  FILLER REDEFINES VNAM4F.
000036              03 VNAM4A    PICTURE X.
000037            02  VNAM4I  PIC X(20).
000038            02  NNAM4L    COMP  PIC  S9(4).
000039            02  NNAM4F    PICTURE X.
000040            02  FILLER REDEFINES NNAM4F.
000041              03 NNAM4A    PICTURE X.
000042            02  NNAM4I  PIC X(20).
000043        01  MAP218O REDEFINES MAP218I.
000044            02  FILLER PIC X(12).
000045            02  FILLER PICTURE X(3).
000046            02  VNAM1O  PIC X(20).
000047            02  FILLER PICTURE X(3).
000048            02  NNAM1O  PIC X(20).
000049            02  FILLER PICTURE X(3).
000050            02  VNAM2O  PIC X(20).
000051            02  FILLER PICTURE X(3).
000052            02  NNAM2O  PIC X(20).
000053            02  FILLER PICTURE X(3).
000054            02  VNAM3O  PIC X(20).
000055            02  FILLER PICTURE X(3).
000056            02  NNAM3O  PIC X(20).
000057            02  FILLER PICTURE X(3).
000058            02  VNAM4O  PIC X(20).
000059            02  FILLER PICTURE X(3).
000060            02  NNAM4O  PIC X(20).
```

Dies ist der Code von "PRAK218.LIB(MSET218)". Er erstreckt sich über mehrere Panels. Wir verwenden es als Vorlage (Template) für die von uns als COBOL-Programm zu erstellende Business Logic.

3. Erstellen des Business Logik Cobol Programms

Wir rufen erneut den Edit-Entry-Panel auf.

```
Menu  RefList  RefMode  Utilities  Workstation  Help
 ------------------------------------------------------------------------------
                               Edit Entry Panel

 ISPF Library:
    Project . . . PRAK218
    Group . . . . CICSDB2  . . .          . . .          . . .
    Type  . . . . COB
    Member  . . . COB218          (Blank or pattern for member selection list)

 Other Partitioned, Sequential or VSAM Data Set:
    Data Set Name . . .
    Volume Serial . . .           (If not cataloged)

 Workstation File:
    File Name . . . . .
                                          Options
 Initial Macro  . . . .                      Confirm Cancel/Move/Replace
 Profile Name . . . . .                      Mixed Mode
 Format Name  . . . . .                      Edit on Workstation
 Data Set Password  . .                      Preserve VB record length

 Command ===>
  F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
 F10=Actions  F12=Cancel
```

Abbildung 5: Anlegen des Members "CPROG218"

Wir legen ein weiteres Member "PRAK218.CICSDB2.COB(COB218)" an (s. Abbildung 8). Es soll unser Business Logic-Programm aufnehmen.

Wir bestätigen mit der Eingabetaste.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
------------------------------------------------------------------------------
EDIT       PRAK218.CICSDB2.COB(COB218) - 01.00            Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100        IDENTIFICATION DIVISION.
000200        PROGRAM-ID. COB218.
000300        ENVIRONMENT DIVISION.
000400        DATA DIVISION.
000500        WORKING-STORAGE SECTION.
000600            EXEC SQL
000700               INCLUDE SQLCA
000800            END-EXEC.
000900        01  NAME-TAB.
001000            02 VORNAME     PICTURE  X(20).
001100            02 NACHNAME    PICTURE  X(20).
001200        COPY MSET218.
001300        LINKAGE SECTION.
001400        PROCEDURE DIVISION.
001500            MOVE LOW-VALUES TO MAP218O.
001600            EXEC SQL
001700              DECLARE C1 CURSOR FOR
001800              SELECT VNAME,NNAME FROM PRAK218.TAB218
001900            END-EXEC.
002000
002100            EXEC SQL  OPEN C1  END-EXEC.
002200
002300            EXEC SQL  FETCH C1 INTO :VORNAME, :NACHNAME   END-EXEC.
002400            MOVE VORNAME TO VNAM1I.
002500            MOVE NACHNAME TO NNAM1I.
002600
002700            EXEC SQL  FETCH C1 INTO :VORNAME, :NACHNAME   END-EXEC.
002800            MOVE VORNAME TO VNAM2I.
002900            MOVE NACHNAME TO NNAM2I.
003000
003100            EXEC SQL  FETCH C1 INTO :VORNAME, :NACHNAME   END-EXEC.
003200            MOVE VORNAME TO VNAM3I.
003300            MOVE NACHNAME TO NNAM3I.
Command ===>                                                 Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Abbildung 6: Der erste Teil des Business Logic-Programms

Dies ist das vollständige COBOL-Programm nach Fertigstellung. Es umfasst mehrere Panels. Mit den F8- bzw. F7-Tasten „scrollen“ wir zwischen den beiden Panels hin und her.

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAK218.CICSDB2.COB(COB218) - 01.00            Columns 00001 00072
003400
003500            EXEC SQL  FETCH C1 INTO :VORNAME, :NACHNAME   END-EXEC.
003600            MOVE VORNAME TO VNAM4I.
003700            MOVE NACHNAME TO NNAM4I.
003800
003900            EXEC SQL  CLOSE C1   END-EXEC.
004000
004100            EXEC CICS SEND MAP('MAP218')
004200                           MAPSET('MSET218')
004300                           ERASE
004400            END-EXEC.
004500
004600            GOBACK.
****** **************************** Bottom of Data ****************************

Command ===>                                                 Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Abbildung 7: Der zweite Teil des Business Logic-Programms

Während der Übersetzung des Mapsets PRAK218.CICSDB2.COB(COBMAP5) wurde ein Member "MSET218" im Dataset "PRAK218.LIB" erstellt, der ein Template für unser COBOL-Programm enthält.

Zeile 1200 unseres COBOL-Programms enthält das Statement

COPY MSET218.

und Zeile 1500

MOVE LOW-VALUES TO MAP218O.

Dieses lädt das vorher angelegte Template automatisch in das COBOL-Programm.

Nach Fertigstellung des Programms kehren wir zum Edit-Entry-Panel zurück.

<u>Aufgabe:</u> ***Erstellen Sie den Member und schreiben Sie das COBOL-Programm hinein. Benutzen Sie als Membernamen COB<Ihre Prak-ID>.***

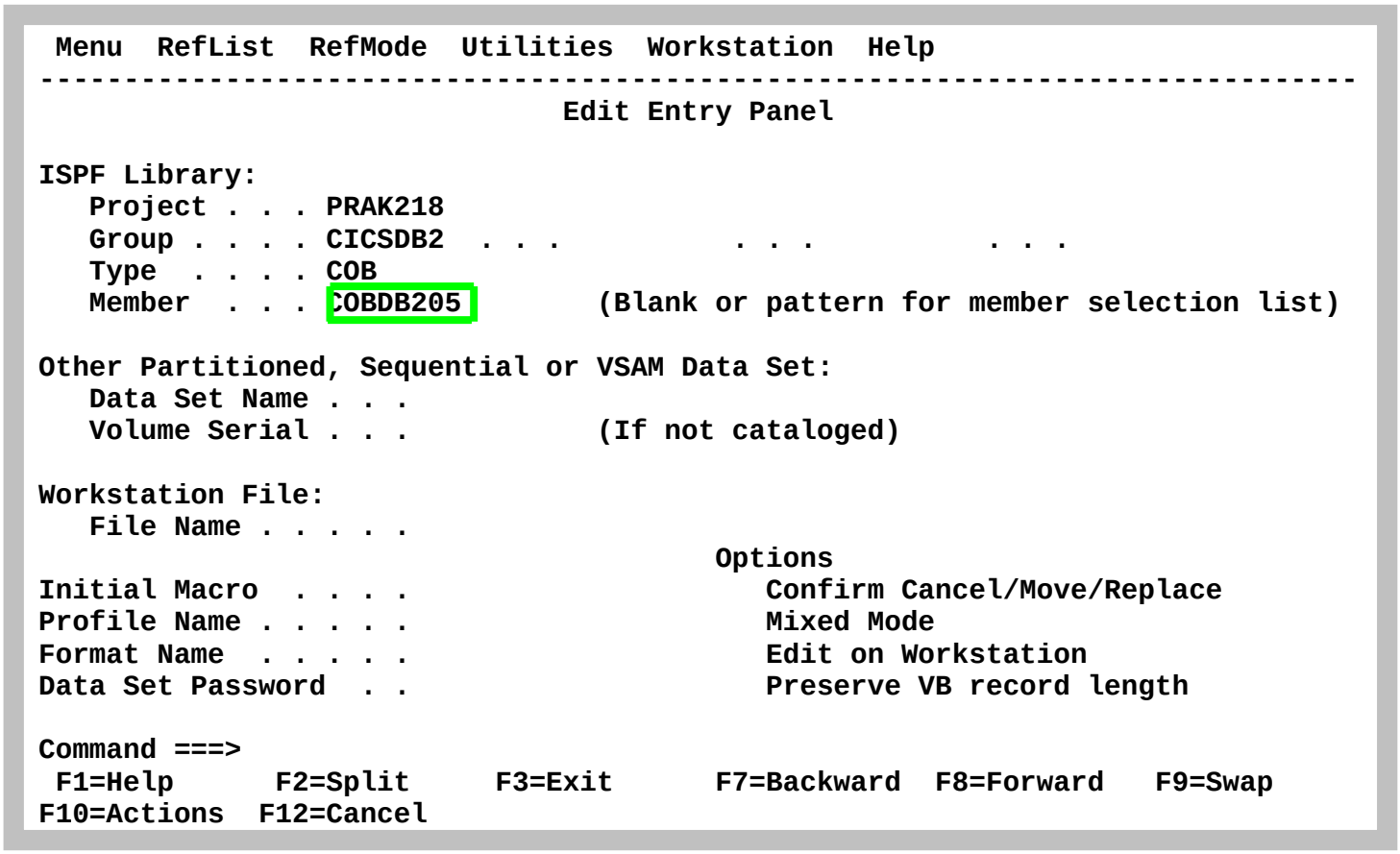

Abbildung 8: Anlegen des Members PCOMPJCL

4. Übersetzung des Cobol Programms

Ehe dieses Programm mit Hilfe des COBOL-Compilers übersetzt werden kann, sind 2 Precompiler-Läufe erforderlich. Der erste Precompiler-Lauf übersetzt alle EXEC SQL-Statements in native DB2 API-Aufrufe.

Wir erstellen ein neues Member „COBDB205" (s. Abbildung 9) zur Aufnahme eines JCL-Scripts, das den SQL-Precompiler aufruft.

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 -------------------------------------------------------------------------------
 EDIT       PRAK218.CICSDB2.COB(COBDB205) - 01.00          Columns 00001 00072
 ****** ***************************** Top of Data ******************************
 ==MSG> -Warning- The UNDO command is not available until you change
 ==MSG>           your edit profile using the command RECOVERY ON.
 000100 //PRAK218D JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
 000200 //             REGION=4M
 000300 //PCOMP EXEC DB2COB
 000400 //PC.STEPLIB DD DISP=SHR,DSN=SYS1.DSN.V910.SDSNEXIT
 000500 //           DD DISP=SHR,DSN=SYS1.DSN.V910.SDSNLOAD
 000600 //DBRMLIB  DD  DSN=PRAK218.DBRMLIB.DATA(COB218),DISP=SHR
 000700 //SYSCIN   DD  DSN=PRAK218.CICSDB2.COB(PCOMPOUT),DISP=SHR
 000800 //SYSLIB   DD  DSN=PRAK218.CICSDB2.COB,DISP=SHR
 000900 //SYSIN    DD  DSN=PRAK218.CICSDB2.COB(COB218),DISP=SHR
 ****** **************************** Bottom of Data ****************************




 Command ===>                                                 Scroll ===> PAGE
  F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Abbildung 9: Das JCL-Script

Wir erstellen das in Abbildung 9 dargestellte JCL-Script zum Aufruf des EXEC SQL-Precompilers.

"PRAK218.DBRMLIB.DATA" ist bis jetzt noch leer. Nach der Ausführung des JCL-Scriptes hat der Precompiler einen Member "PRAK218.DBRMLIB.DATA(COB218)" angelegt.

Auf der Kommandozeile geben wir wieder den SUBMIT-Befehl "SUB" ein. Wir warten die Ausführung des JES-Jobs ab (s. Abbildung 10) und bestätigen diese dann mit der Eingabetaste.

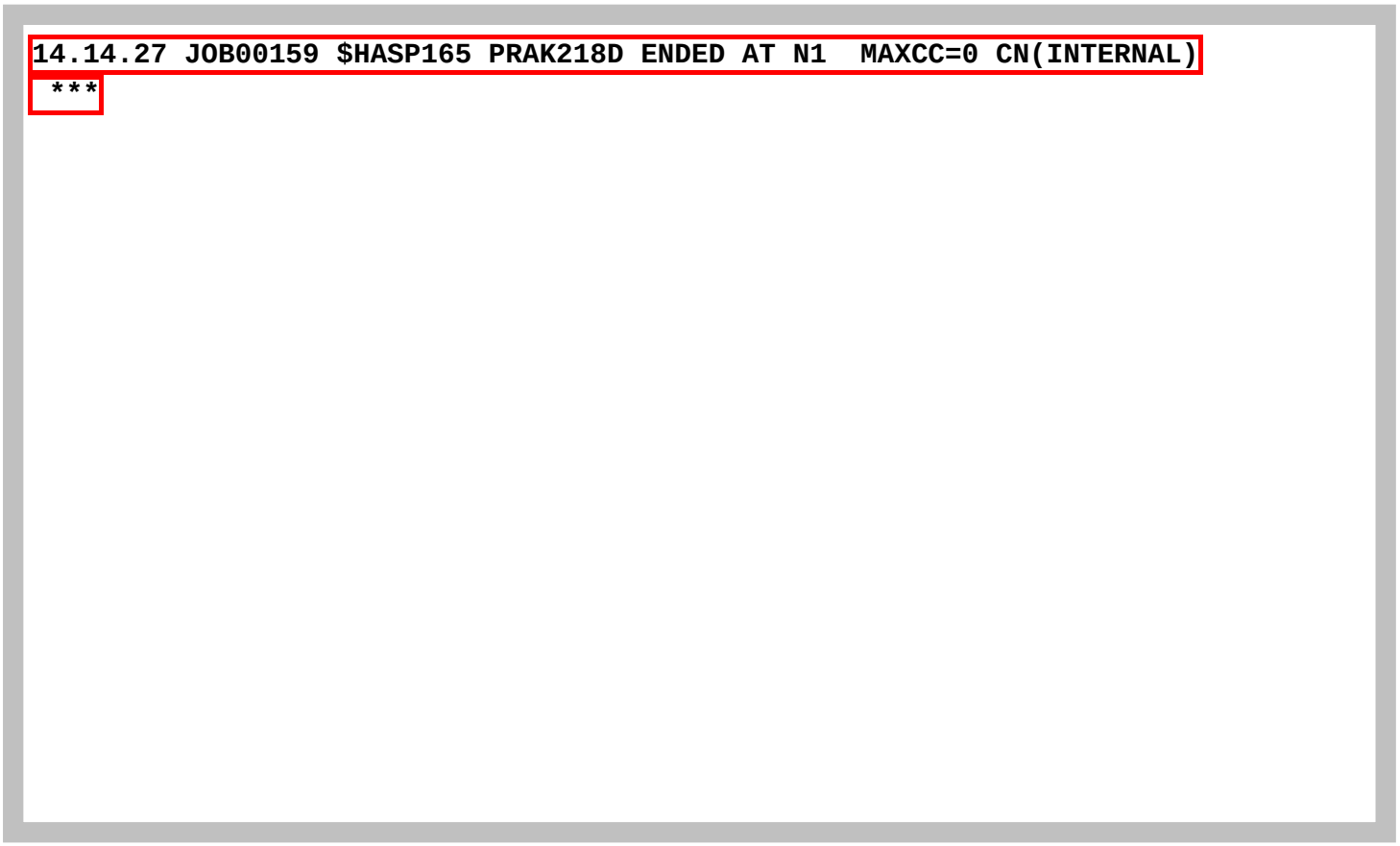

Abbildung 10: Bestätigung der Jobverarbeitung

"MAXCC=0" oder "MAXCC=4" zeigt an, dass der Befehl erfolgreich ausgeführt wurde. Wir bestätigen mit der Eingabetaste.

Aufgabe: ***Erstellen Sie einen neuen Member und schreiben Sie das JCL-Script, das den Precompiler-Aufruf enthält, hinein. Führen Sie es anschließend aus. Denken Sie daran, den Jobnamen ' PRAK218D ' wieder an Ihren Mainframe-Accountnamen anzupassen.***

Als nächstes erstellen wir einen neuen Member "COBSTA05" (s. Abbildung 11) zur Aufnahme eines JCL-Scripts, das folgende Funktionen aufruft:

- **den CICS-Precompiler**
- **den COBOL-Compiler**
- **den Linker**

Anschließend betätigen wir die Eingabetaste.

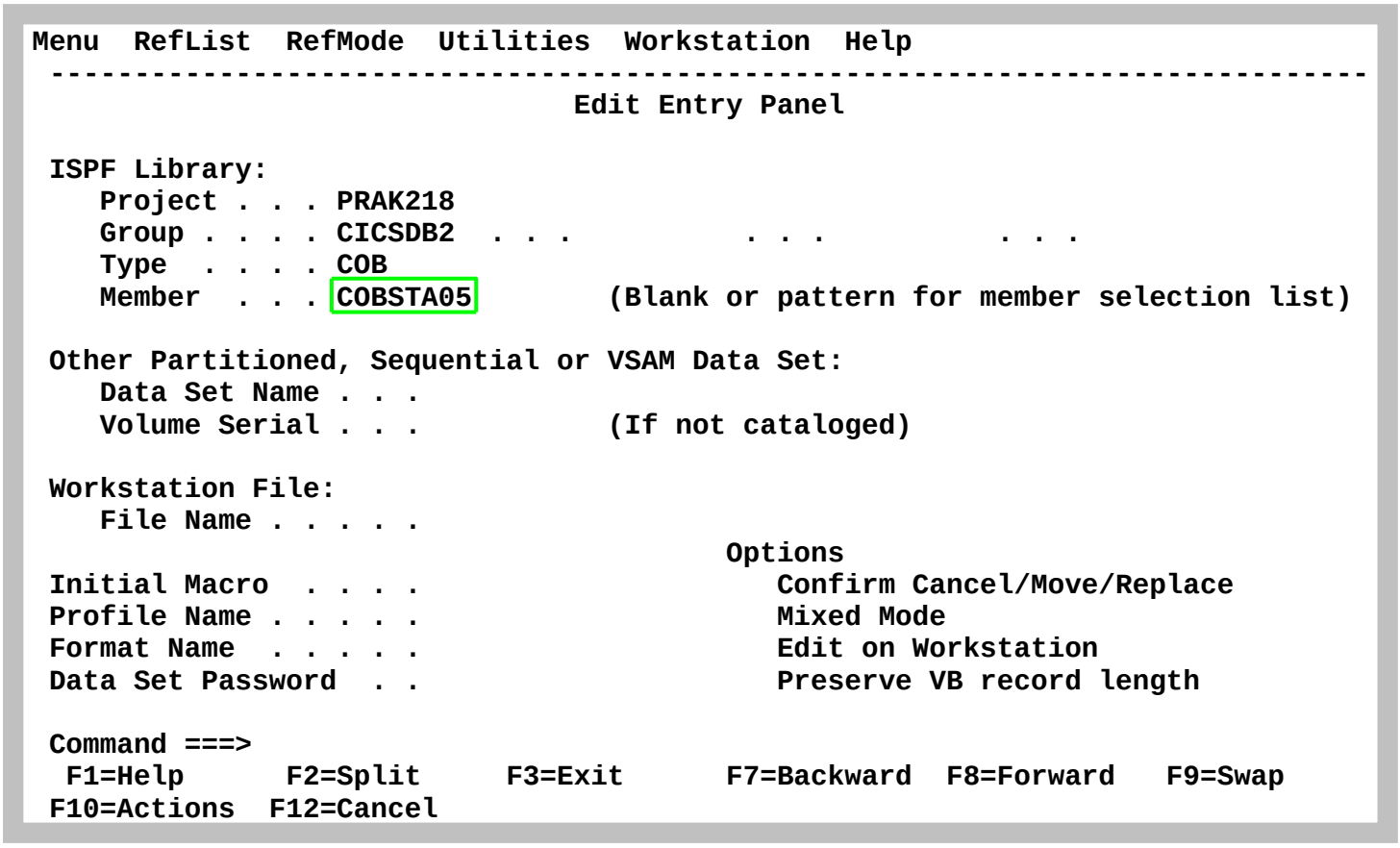

Abbildung 11: Anlegen des Members "COBSTA05"

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       PRAK218.CICSDB2.COB(COBSTA05) - 01.05           Columns 00001 00072
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000100 //PRAK218C JOB (),CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=&SYSUID,
000200 //         REGION=4M,LINES=10,CARDS=1000
000300 //COMP   EXEC COBCICS,PARM.TRN='COBOL3'
000400 //TRN.SYSIN DD DISP=SHR,DSN=PRAK218.CICSDB2.COB(PCOMPOUT)
000500 //COB.SYSLIB DD DSN=PRAK218.LIB,DISP=SHR
000600 //LKED.SYSIN DD *
000700  INCLUDE SYSLIB(DSNCLI)
000800  NAME COB218(R)
001000 //BIND     EXEC PGM=IKJEFT01
001100 //STEPLIB  DD DISP=SHR,DSN=DSN931.DSN.V910.SDSNEXIT
001200 //         DD DISP=SHR,DSN=SYS1.DSN.V910.SDSNLOAD
001300 //DBRMLIB  DD DISP=OLD,DSN=PRAK218.DBRMLIB.DATA(COB218)
001400 //SYSPRINT DD SYSOUT=*
001500 //SYSTSPRT DD SYSOUT=*
001600 //SYSUDUMP DD SYSOUT=*
001700 //SYSTSIN  DD  *
001800   DSN S(D931)
001900   BIND  PLAN(ZGR218CO) MEMBER(COB218) ACTION(REP) RETAIN ISOLATION(CS)
002000   END
002200 //GRANT    EXEC PGM=IKJEFT01
002300 //STEPLIB  DD DISP=SHR,DSN=SYS1.DSN.V910.SDSNLOAD
002400 //SYSPRINT DD SYSOUT=*
002500 //SYSTSPRT DD SYSOUT=*
002600 //SYSUDUMP DD SYSOUT=*
002700 //SYSTSIN  DD *
002800   DSN SYSTEM(D931)
002900   RUN PROGRAM(DSNTIAD) PLAN(DSNTIA91) -
003000       LIBRARY('SYS1.DSN.V910.SDSNLOAD')
003100   END
003200 //SYSIN    DD *
003300 GRANT EXECUTE ON PLAN ZGR218CO TO PUBLIC
003400 /*
****** *************************** Bottom of Data ****************************
Command ===>                                                 Scroll ===> PAGE
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Abbildung 12: Das JCL-Script (Panel #1)

Der SQL-Precompiler-Lauf hat im Dataset "PRAK218.DBRMLIB.DATA" ein Member "COB218" angelegt (Zeile 800).
Die Ausführung unseres Programms "COB218" unter CICS benötigt einen Zeiger auf die anzusprechende Datenbank-Tabelle (als im JCL-Script als "PLAN" bezeichnet). Wir geben diesem Zeiger den Namen "ZGR218CO" (Zeile 1900 und 3300).

Wir geben "SUB" auf der Kommandozeile ein, warten, bis JES den Job ausgegeben hat und bestätigen anschließend mit der Eingabetaste.

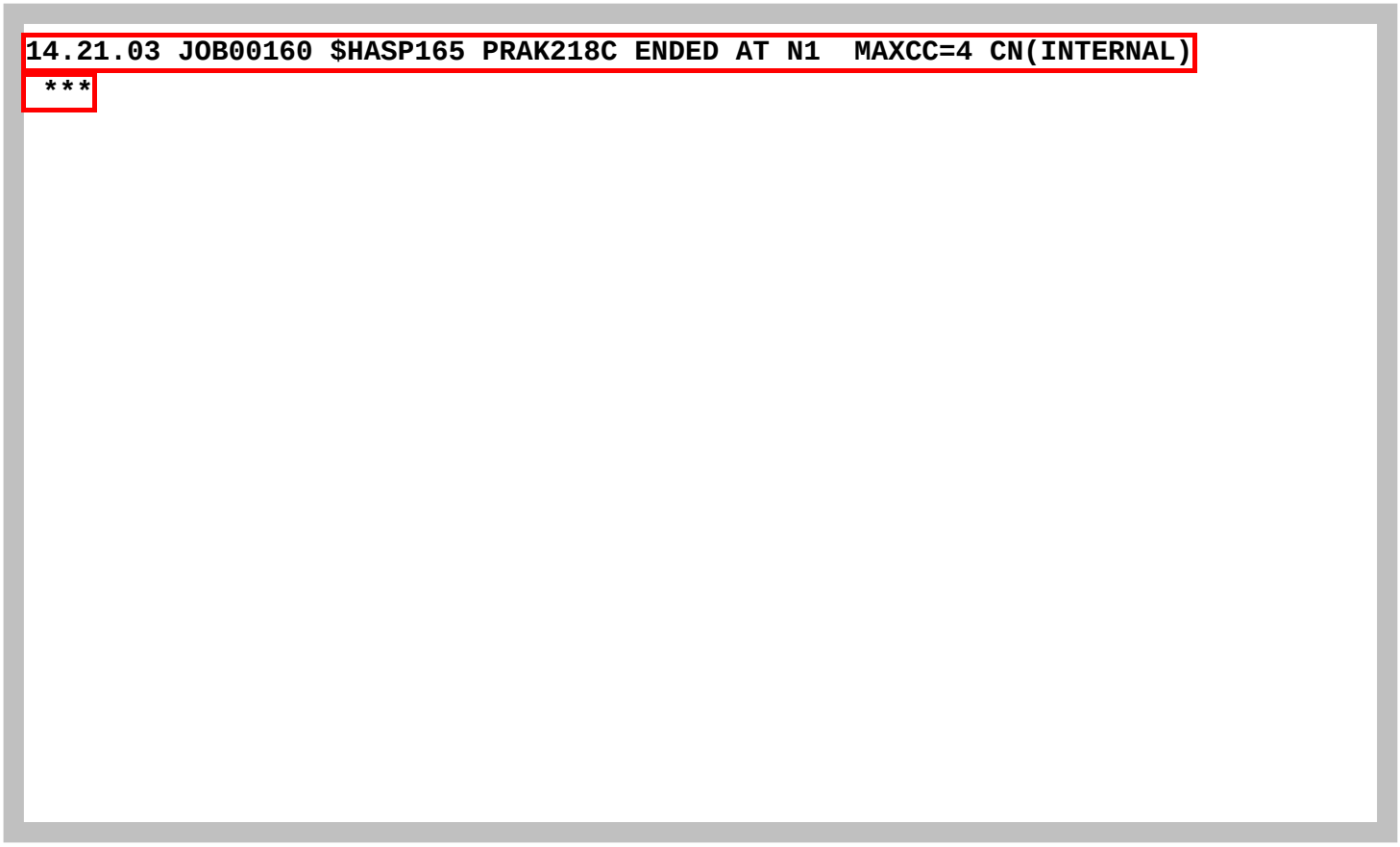

Abbildung 13: Ausgabe der Jobverarbeitung

"MAXCC=4" bedeutet, dass der Compile- und Link-Lauf erfolgreich durchgeführt wurde.

Aufgabe: *Erstellen Sie einen neuen Member, legen Sie das JCL-Script COBSTA05 an (mit an Ihren Accountnamen angepasstem Jobnamen) und führen Sie es aus. Benutzen Sie als Zeiger (Plan) den Bezeichner ZGR<Ihre Prakt-Nr>CO.*

Wir haben nun alle Programme für unsere CICS - DB2-Transaktion erstellt. Als nächsten Schritt müssen sie in dem CICS-Subsystem installiert werden. Hierzu öffnen wir eine weitere Z/OS-Session.

5. Installation im CICS Subsystem

```
TCPIP MSG10 ==> SOURCE DATA SET = SYS1.LOCAL.VTAMLST(USSTCPIP)
z/OS Z18 Level 0609                                IP Address = 134.2.212.63
                                                    VTAM Terminal = SC0TCP18

                      Application Developer System

                               //  OOOOOOO   SSSSS
                              //  OO    OO SS
                      zzzzzz //  OO    OO SS
                        zz  //  OO    OO SSSS
                      zz   //  OO    OO     SS
                    zz    //  OO    OO     SS
                  zzzzzz //   OOOOOOO  SSSS


                  System Customization - ADCD.Z18.*




 ===> Enter "LOGON" followed by the TSO userid. Example "LOGON IBMUSER" or
 ===> Enter L followed by the APPLID
 ===> Examples: "L TSO", "L CICS", "L IMS3270

L CICS
```

Abbildung 14a: Der Login-Screen

Wir loggen uns mit "L CICS" ein (s. Abbildung 14a) und bestätigen mit der Eingabetaste.

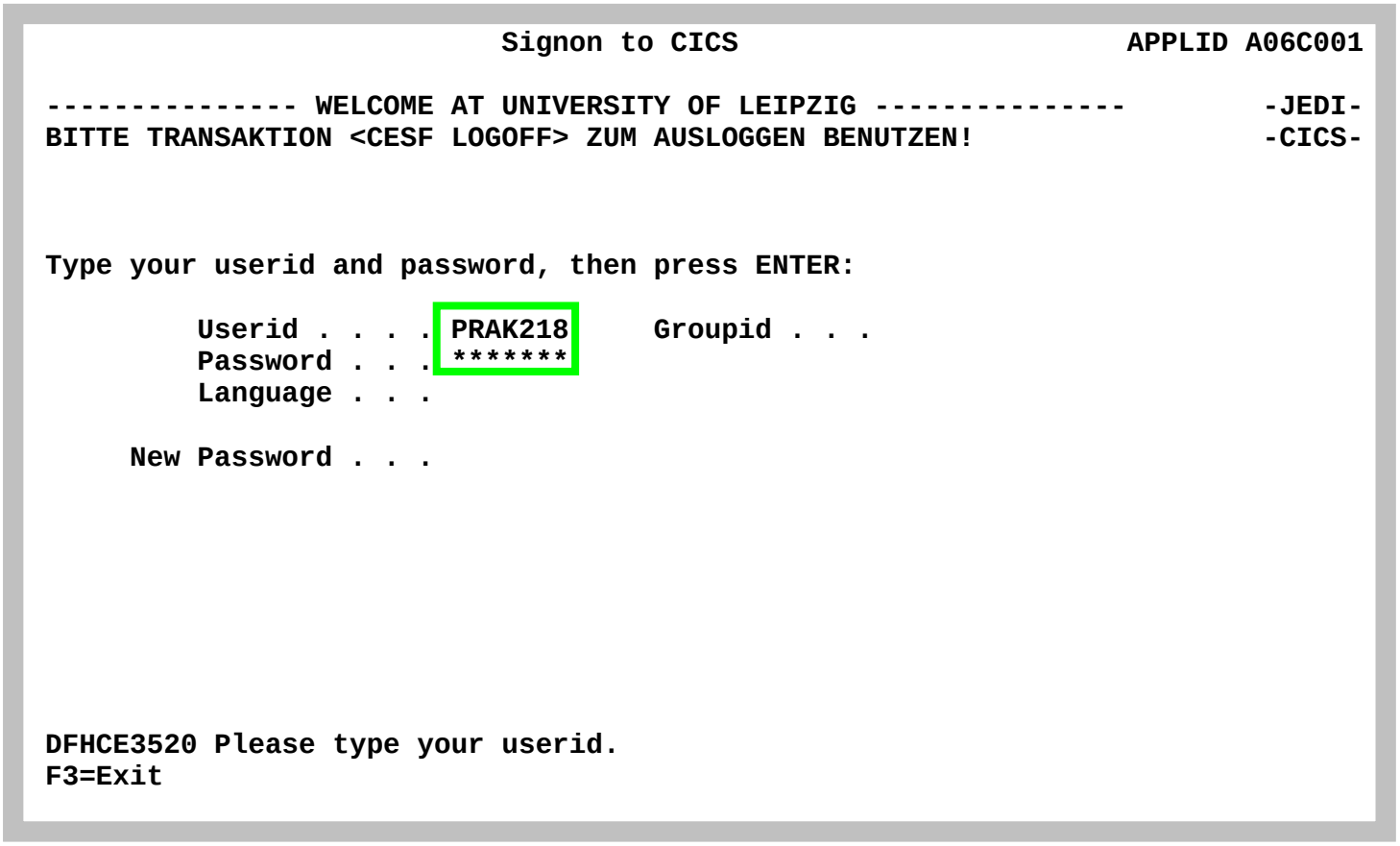

Abbildung 14b: Signon to CICS-Screen

Wir müssen uns unter CICS mit der gleichen Userid wie unter TSO einloggen (s. Abbildung 14b). Auch unser TSO-Password ist in dieses Panel einzugeben. Durch das Betätigen der Eingabetaste kommen wir in den nächsten Screen.

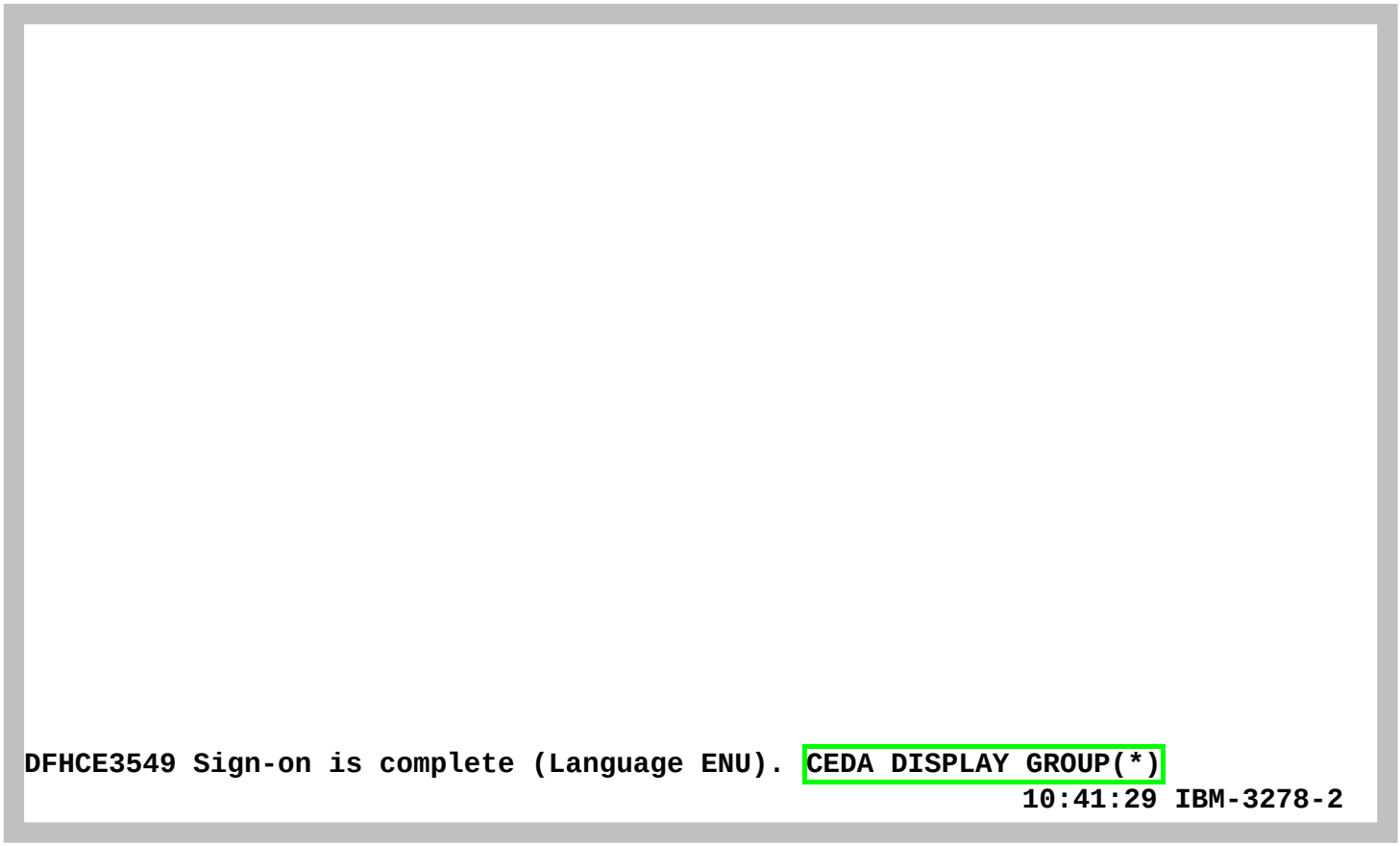

Abbildung 15: Einloggvorgang ist abgeschlossen

Wir betätigen die Tab-Taste, so dass der Cursor auf die letzte Zeile springt (Abbildung 15). Hier geben wir den "CEDA DISPLAY GROUP(*)"-Befehl ein und bestätigen anschließend mit der Eingabetaste.

Der Group-Name kann beliebig gewählt, aber immer nur einmal vergeben werden. Der Übersichtlichkeit wegen ist es sinnvoll, den Login-Namen zu verwenden. Wir haben aber bereits die Gruppe PRAK218 in Tutorial 3 verwendet. Wir löschen deshalb die Gruppe PRAK218 mit dem Befehl

CEDA DELETE ALL GROUP(PRAK218)

Und verifizieren danach mit CEDA DISPLAY GROUP(*) dass dies auch tatsächlich geschehen ist.

An dieser Stelle ist es jetzt notwendig, eine neue Gruppe anzulegen. Wir definieren zunächst unser BMS-Programm mit dem Namen "MSET218" für die neue Group "PRAK218" und betätigen anschließend dreimal die Eingabetaste.

```
 CEDA DEFINE MAPSET(MSET218) GROUP(PRAK218)
 OVERTYPE TO MODIFY                                      CICS RELEASE = 0640
  CEDA  DEFine Mapset( MSET218  )
   Mapset         : MSET218
   Group          : PRAK218
   Description  ==>
   REsident     ==> No                 No | Yes
   USAge        ==> Normal             Normal | Transient
   USElpacopy   ==> No                 No | Yes
   Status       ==> Enabled            Enabled | Disabled
   RSl            : 00                 0-24 | Public









   I New group PRAK218 created.
                                                  SYSID=CICS APPLID=CICS
   DEFINE SUCCESSFUL                          TIME:  14.27.04  DATE: 12.120
PF 1 HELP 2 COM 3 END           6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 16: Definition des Mapsets "SET5218"

Die Definition war erfolgreich und die neue Gruppe wurde erstellt.

Als nächstes wird das COBOL-Programm definiert. Dazu drücken wir die Eingabetaste.

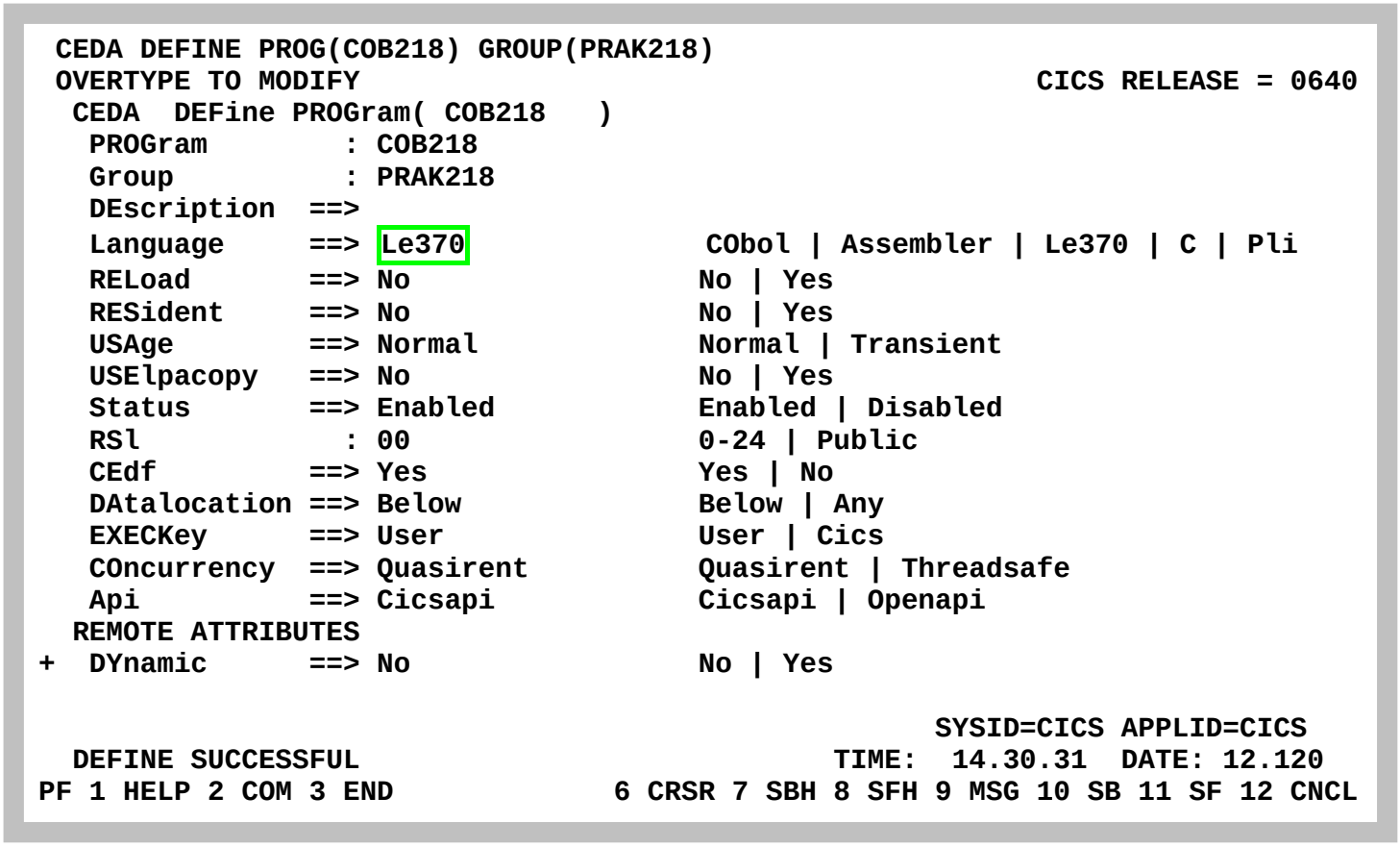

Abbildung 17: Auswahl der Parameter

Le370 wird bei Language eingeben; sie ist aber eigentlich eine Entwicklungsumgebung (s. Abbildung 17)

Auch hier bestätigen wir mit der Eingabetaste.

Die Nachricht "DEFINE SUCCESSFUL" erscheint; wir beenden diese Aktion mit Betätigung der F3-Taste.

```
CEDA DEFINE TRANS(X218) GROUP(PRAK218)
  STATUS:  SESSION ENDED
```

Abbildung 18: Sitzung beendet

Als letztes müssen wir die Bezeichnung der neuen Transaktion definieren. Wir wählen auch hierfür den Namen "X218". Der Namen der Transaktion muss immer aus vier Zeichen bestehen. Man könnte den Namen auch anders wählen. Für die Teilnehmer sind die Rechte jedoch so vergeben, dass die Transaktion nur wie unten angegeben angelegt werden kann. Wir geben das Kommando "CEDA DEFINE TRANS(X218) GROUP(PRAK218)" ein (s. Abbildung 18) und bestätigen mit der Eingabetaste.

In die Zeile "PROGram" geben wir nun "COB218" (Abbildung 19) ein und bestätigen dies mit der Eingabetaste.

```
 DEFINE TRANS(X218) GROUP(PRAK218)
 OVERTYPE TO MODIFY                                      CICS RELEASE = 0640
  CEDA  DEFine TRANSaction( X218 )
   TRANSaction  ==> X218
   Group        ==> PRAK218
   DEscription  ==>
   PROGram      ==> COB218
   TWasize      ==> 00000             0-32767
   PROFile      ==> DFHCICST
   PArtitionset ==>
   STAtus       ==> Enabled           Enabled | Disabled
   PRIMedsize     : 00000             0-65520
   TASKDATALoc  ==> Below             Below | Any
   TASKDATAKey  ==> User              User | Cics
   STOrageclear ==> No                No | Yes
   RUnaway      ==> System            System | 0 | 500-2700000
   SHutdown     ==> Disabled          Disabled | Enabled
   ISolate      ==> Yes               Yes | No
   Brexit       ==>
+ REMOTE ATTRIBUTES
  S PROGRAM OR REMOTESYSTEM MUST BE SPECIFIED.
                                                    SYSID=CICS APPLID=CICS

PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 19: Definition der Transaktion

"DEFINE SUCCESSFUL" erscheint; also war die Definition erfolgreich, wir beenden sie mit der F3-Taste.

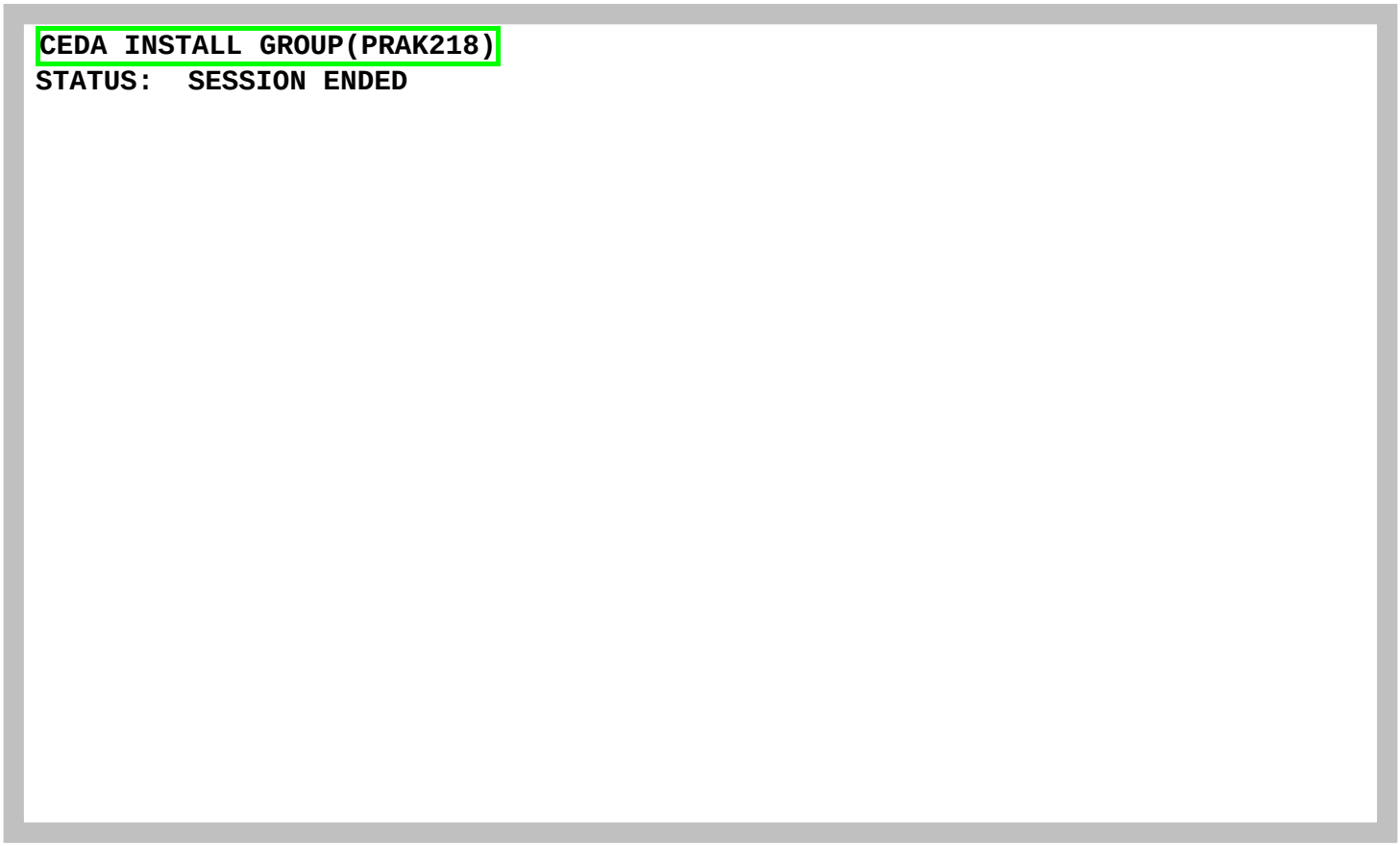

Abbildung 20: Installation der Gruppe

Nachdem die BMS-MAP, das COBOL-Programm und die Transaktionsbezeichnung definiert worden sind, wird nun alles in unserer Gruppe "PRAK218" installiert. Dazu geben wir den Befehl "CEDA INSTALL GROUP(PRAK218)" ein (s. Abbildung 21) und bestätigen mit der Eingabetaste.

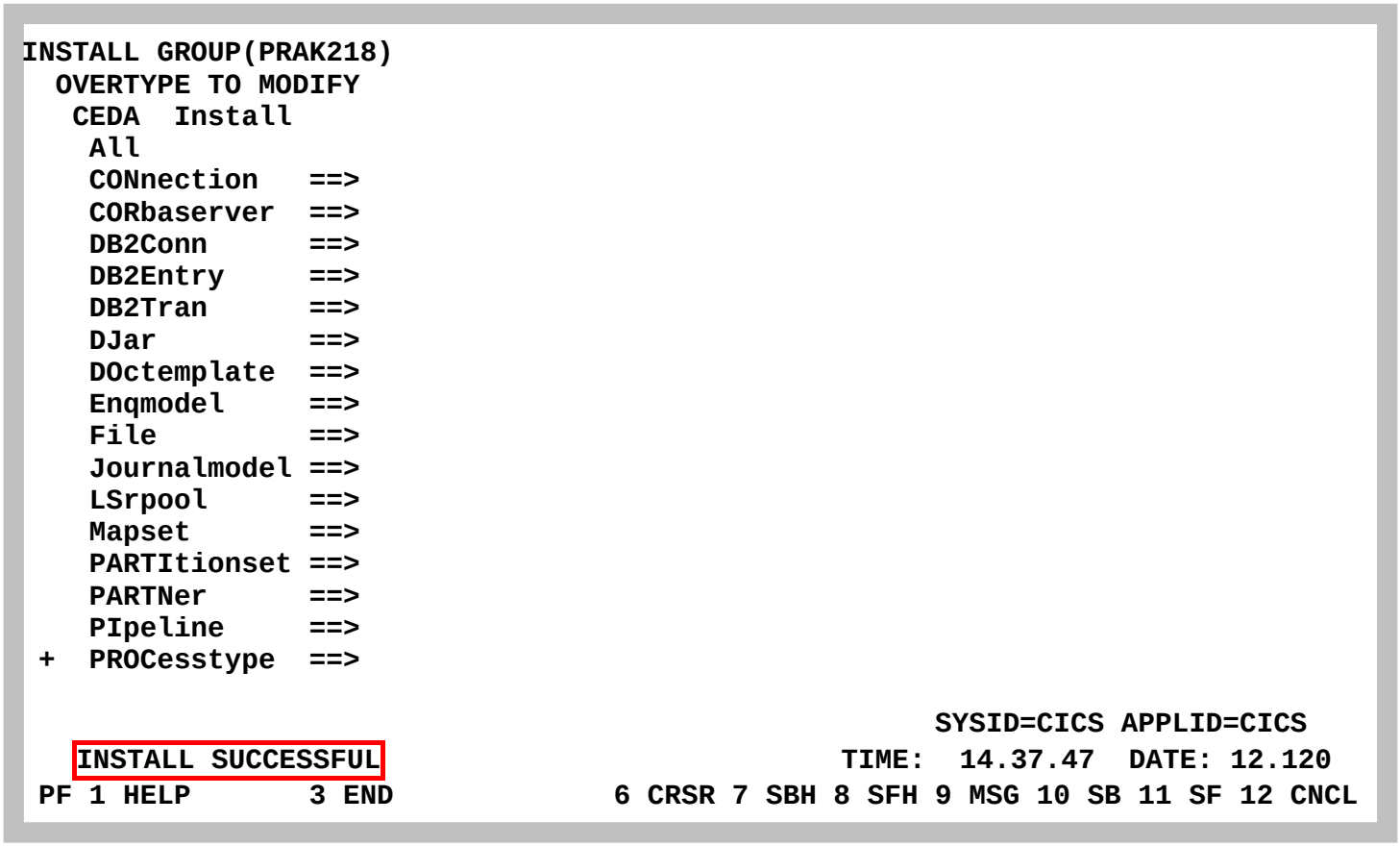

Abbildung 21: Installation war erfolgreich

Die erfolgreiche Installation der Gruppe "PRAK218" zeigt die Ausgabe "INSTALL SUCCESSFUL" (s. Abbildung 21) an. Wir beenden diese Installation, indem wir die F3-Taste drücken.

Es kann sein, dass an dieser Stelle die Meldung „Install unsuccessful“ erscheint. Eine Erläuterung dieses Fehlers finden Sie im Anhang dieses Tutorials.

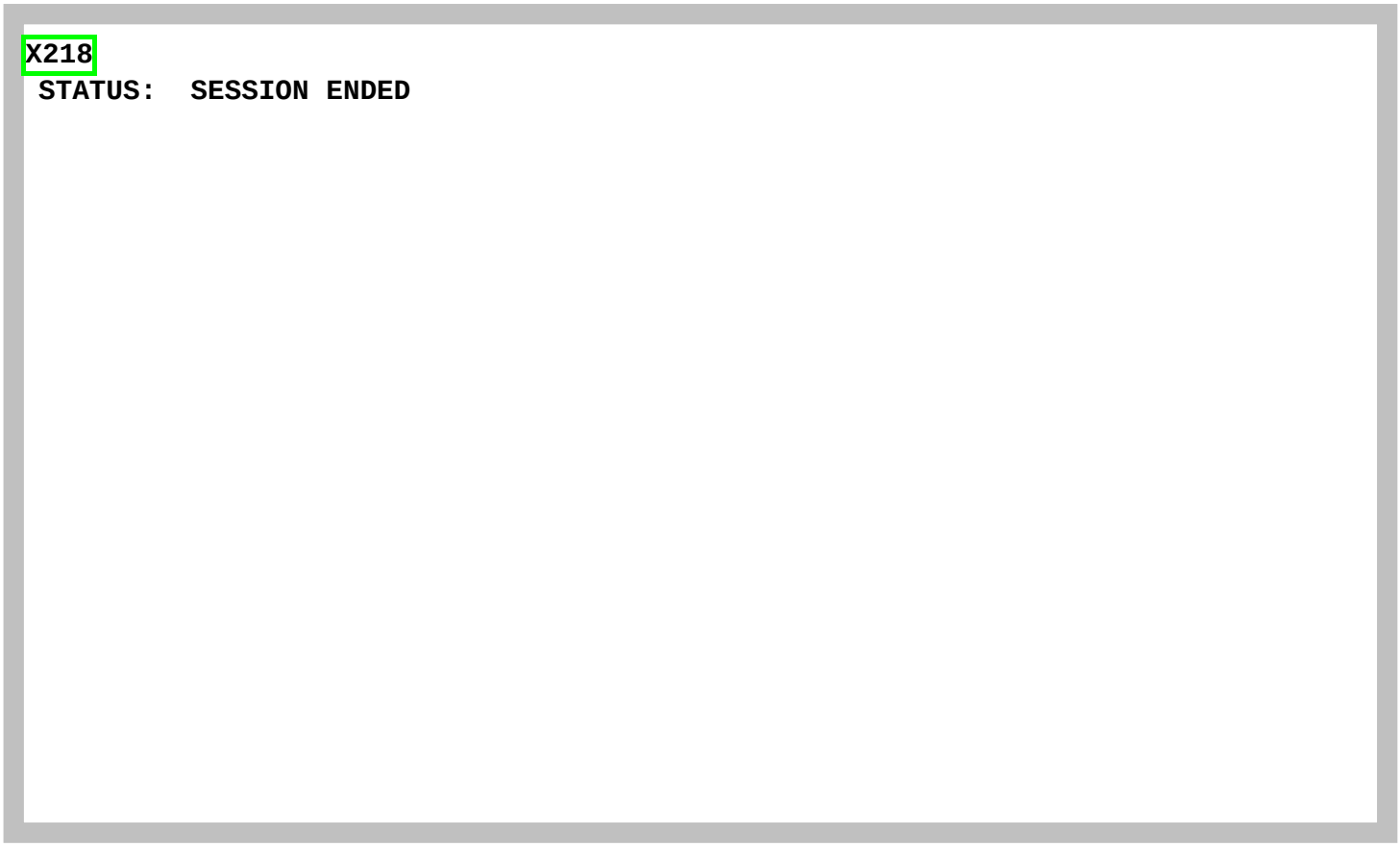

Abbildung 22: Aufruf der Transaktion

In Tutorial 3 waren wir mit der Definition und Installation unserer Transaktion fertig. Wir versuchen es einmal, indem wir unsere Transaktion mit der Bezeichnung "X218" aufrufen. Dazu tragen wir den Namen in die CICS-Kommandozeile ein (s. Abbildung 22) und bestätigen mit der Eingabetaste.

```
DFHAC2220 20:49:29 A06C001 The coordinator system has indicated that the
  current unit of work is to be backed out. Transaction X218 has been
  abnormally terminated with abend ASP3.
```

Abbildung 23: Fehlermeldung

Wir erhalten eine Fehlermeldung (s. Abbildung 23).
Manchmal erscheint auch eine andere Fehlermeldung als die in Abbildung 23 dargestellte.

Die Beschreibung zur Fehlermeldung ASP3 findet sich im CICS IBM Online-Handbuch mit dem folgenden Eintrag:

> **Explanation: The abnormal termination occurs because a remote system on which the unit of work depends fails to take a syncpoint. The transaction cannot commit its changes until all coupled systems to which function has been transmitted also commit. This may be because the syncpoint protocol for transaction to transaction has been violated by failing to be in send mode for all sessions for which syncpoint has not been received.**
> **User Response:**
>
> **Check why the remote system failed to respond to the request.**

TSO, CICS und DB2 sind unterschiedliche z/OS-Subsysteme, die in getrennten virtuellen Adressräumen laufen. Die CICS-Gruppe "PRAK218" benötigt eine Definition unserer Datenbank und Datenbanktabelle.

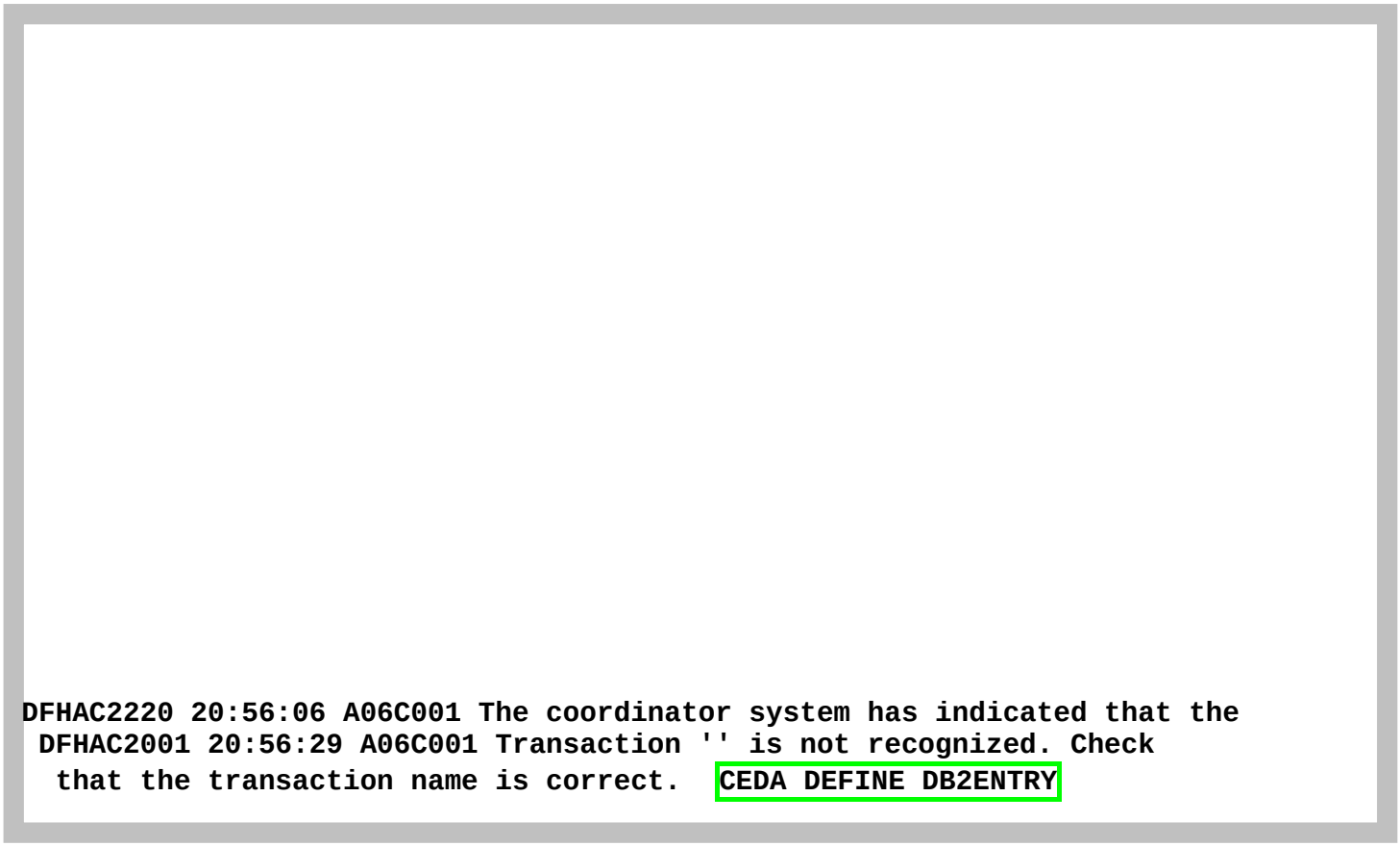

Abbildung 24: Aufruf der Definition der Datenbank

Die Definition erfolgt mit dem Kommando "CEDA DEFINE DB2ENTRY" (s. Abbildung 24).

Es kann auch sein, dass das System sich an dieser Stelle aufhängt. Resultat: keine Tastatureingabe ist möglich, und links unten erscheint ein Strich-Männchen. Drücken der PF2 Taste behebt dies Problem.

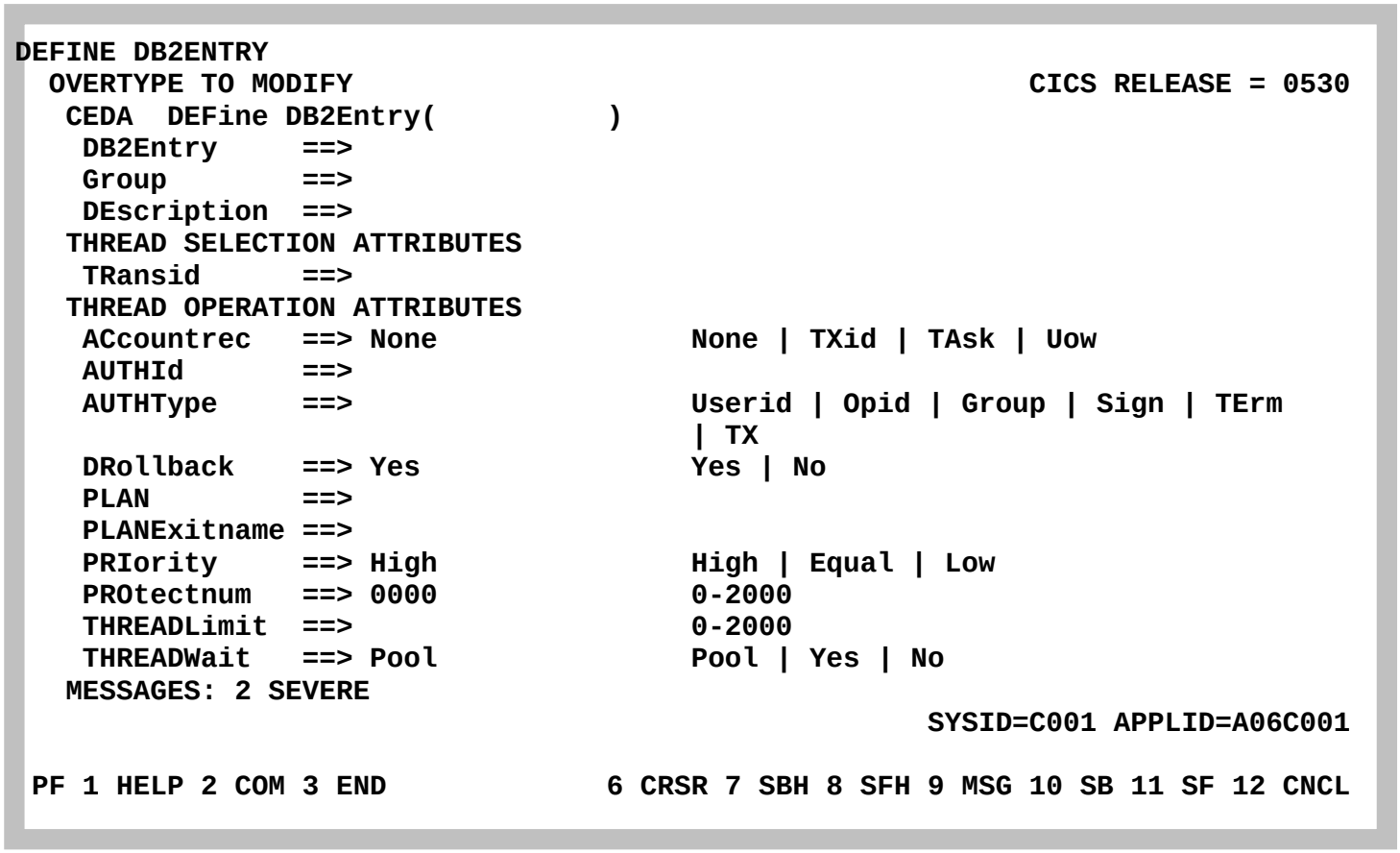

Abbildung 25: DEFINE DB2ENTRY-Panel

Nachdem wir die Eingabetaste gedrückt haben, erscheint der "DEFINE DB2ENTRY-Panel" (s. Abbildung 25). Wir müssen die fehlenden Angaben eintragen und betätigen abschließend die Eingabetaste (s. Abbildung 26).

```
define DB2ENTRY
 OVERTYPE TO MODIFY                                        CICS RELEASE = 0530
  CEDA  DEFine DB2Entry(          )
   DB2Entry     ==> X218
   Group        ==> PRAK218
   DEscription  ==>
  THREAD SELECTION ATTRIBUTES
   TRansid      ==> X218
  THREAD OPERATION ATTRIBUTES
   ACcountrec   ==> TXid             None | TXid | TAsk | Uow
   AUTHId       ==>
   AUTHType     ==> Sign             Userid | Opid | Group | Sign | TErm
                                      | TX
   DRollback    ==> Yes               Yes | No
   PLAN         ==> ZGR218CO
   PLANExitname ==>
   PRIority     ==> High              High | Equal | Low
   PROtectnum   ==> 0000              0-2000
   THREADLimit  ==> 0003             0-2000
   THREADWait   ==> Yes         Pool | Yes | No
  MESSAGES: 2 SEVERE
                                                  SYSID=C001 APPLID=A06C001

 PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 26: Eingabe der Parameter

Wir bezeichnen den DB2-Zugriff (DB2Entry) mit dem Namen "X218". Das Ganze wird Teil der Gruppe "PRAK218". Unsere Transaction-ID (TRansid) ist "X218". Wir hatten ein JCL-Script "COBSTA05" erstellt, das unser COBOL-Programm übersetzte. In diesem Script definierten wir an zwei Stellen einen Zeiger auf unsere Datenbanktabelle (Plan) mit dem Namen "ZGR218CO". Hier wird jetzt für CICS die Verknüpfung zu der Datenbanktabelle hergestellt.

```
 OVERTYPE TO MODIFY                                          CICS RELEASE = 0530
  CEDA  DEFine DB2Entry( X218     )
   DB2Entry       : X218
   Group          : PRAK218
   DEscription  ==>
  THREAD SELECTION ATTRIBUTES
   TRansid      ==> X218
  THREAD OPERATION ATTRIBUTES
   ACcountrec   ==> TXid              None | TXid | TAsk | Uow
   AUTHId       ==>
   AUTHType     ==> Userid            Userid | Opid | Group | Sign | TErm
                                      | TX
   DRollback    ==> Yes               Yes | No
   PLAN         ==> ZGR218CO
   PLANExitname ==>
   PRIority     ==> High              High | Equal | Low
   PROtectnum   ==> 0000              0-2000
   THREADLimit  ==> 0003              0-2000
   THREADWait   ==> Yes               Pool | Yes | No

                                                  SYSID=C001 APPLID=A06C001
   DEFINE SUCCESSFUL                          TIME:  00.00.00  DATE: 01.061
PF 1 HELP 2 COM 3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Abbildung 27: Bestätigung der gelungenen Definition

Die Definition war erfolgreich und wird bestätigt durch die Ausgabe: "DEFINE SUCCESSFUL" (s. Abbildung 27).

Wir verlassen die Definition mit der F3-Taste.

```
CEDA INSTALL GROUP(PRAK218)
 STATUS:  SESSION ENDED
```

Abbildung 28: Installation der Gruppe

Diese Änderung muss wieder installiert werden. Dazu geben wir wieder den Befehl "CEDA INSTALL GROUP(PRAK218)" (s. Abbildung 28) ein und bestätigen mit der Eingabetaste.

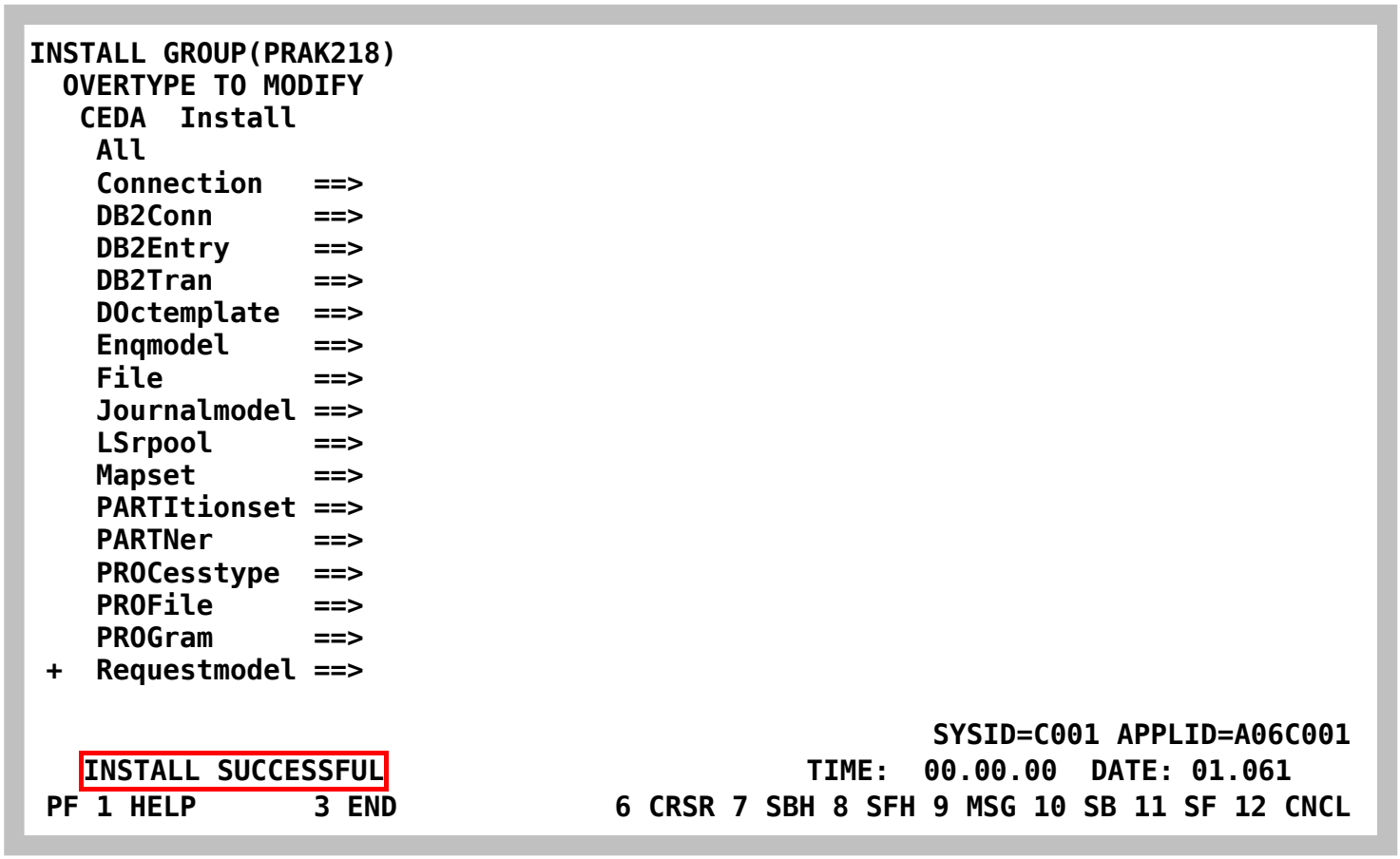

Abbildung 29: Installation der Gruppe

Die Ausgabe „INSTALL SUCCESSFUL“ in der Abbildung 29 sagt aus, dass die Installation erfolgreich war. Wir verlassen diesen Screen wieder mit F3.

6. Ausführen der Transsaktion

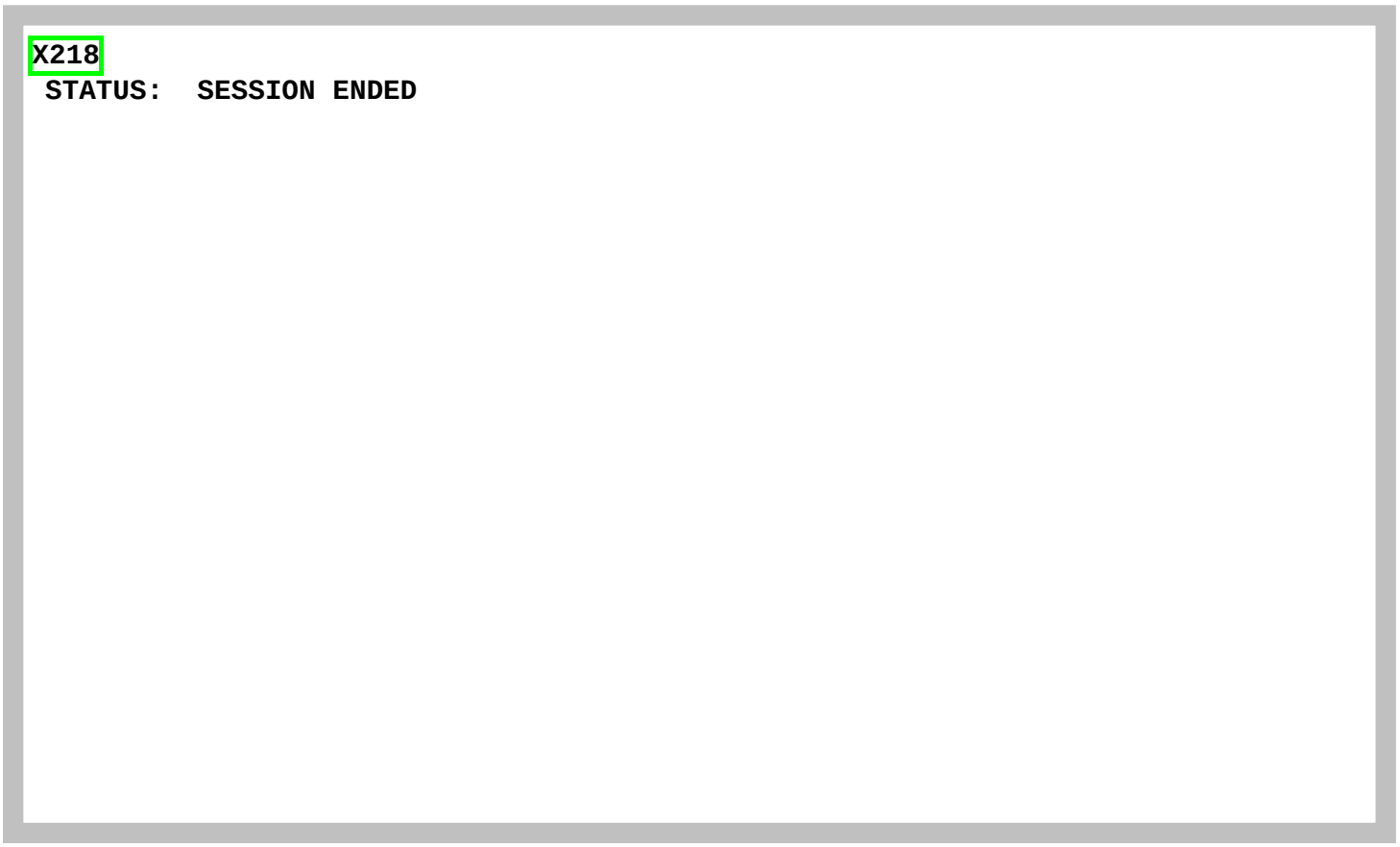

Abbildung 30: Starten der Transaktion

Wir geben den Namen unserer Transaktion "X218" ein, um diese aufzurufen (s. Abbildung 30) und bestätigen mit der Eingabetaste.

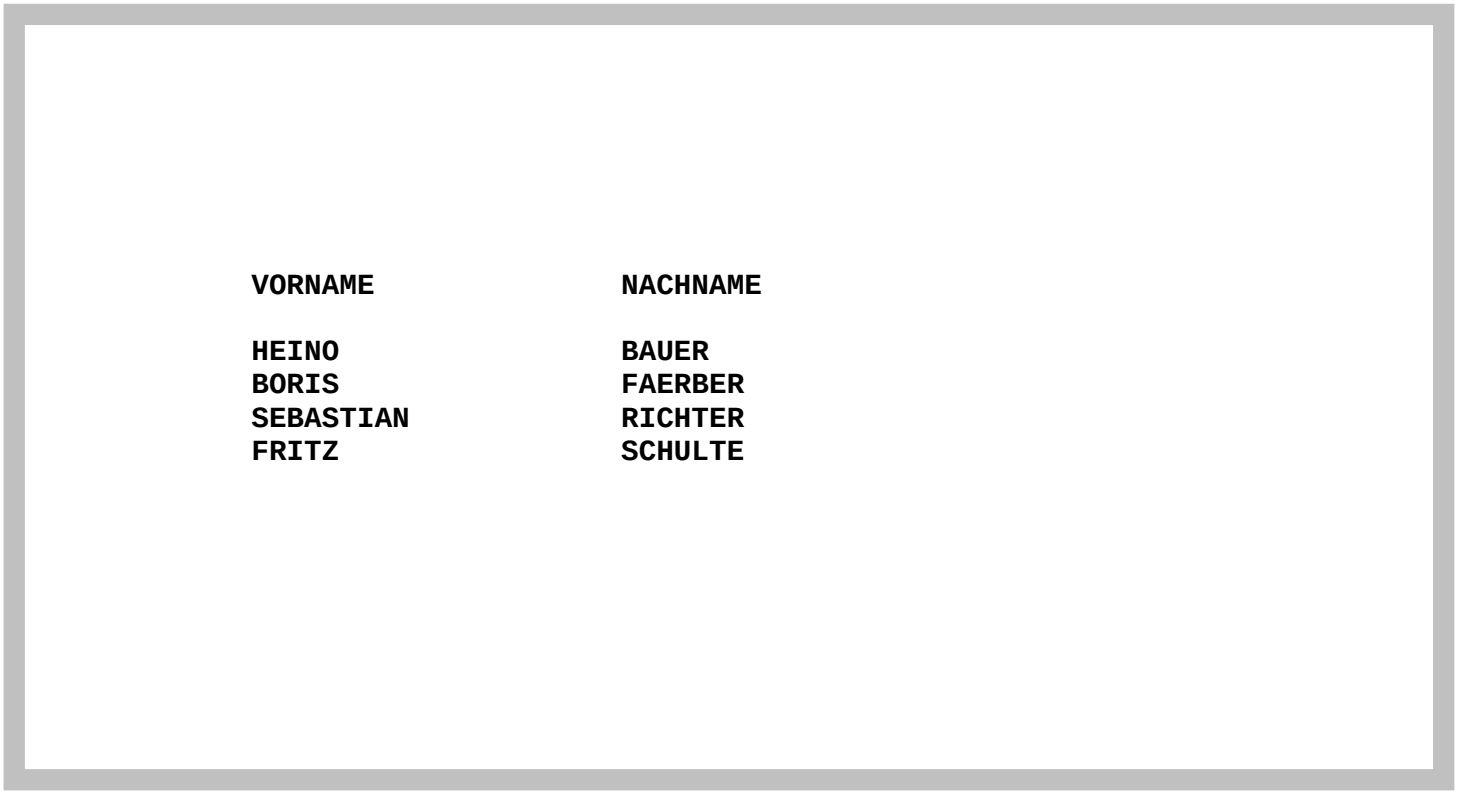

Abbildung 31: Ausgabe der Datenbanktabelle

Die korrekte Ausgabe der Datenbank erscheint auf dem Bildschirm (s. Abbildung 31).

Aufgabe: ***Bereiten Sie unter CICS die Transaktion vor, die auf die DB2-Datenbank zugreifen soll und führen Sie diese anschließend aus. Benutzen Sie dabei als CICS-Gruppen-Namen Ihren Accountnamen, also z.B. PRAK145 oder PRAK162. Die DB2-Datenbank soll Ihren Namen / Ihre Namen enthalten. Benutzen Sie als Transactions-ID "X<Ihre Prakt-ID>". Bezeichnen Sie den DB2ENTRY identisch zu Ihrer Transaction-ID.***

Erzeugen Sie einen Screenshot (unter Windows durch den Shortcut ALT-Druck) Ihrer Version der Abbildung 31 und schicken Sie diesen Ihrem Betreuer per Mail zu. Der Screenshot darf eine Größe von 250 KByte nicht überschreiten, benutzen Sie möglichst das JPG-Format, dass mit Dateigrößen unter 90 KByte auskommt. Löschen Sie nichts von Ihrer Lösung, so dass Ihr Betreuer Ihre Transaktion aufrufen kann.

Aufgabe: ***Gehen Sie vom CUSTOMPAC MASTER APPLICATION MENU aus in die System Display and Search Facility. Im erscheinenden SDSF PRIMARY OPTION MENU wählen Sie die Option ST. Löschen Sie alle angezeigten Jobs, die sich in der PRINT-Queue befinden, indem Sie links neben einen jeden Jobnamen "p" (purge) eintragen und anschließend die Eingabetaste (mehrfach) drücken. Einen Job dürfen Sie natürlich nicht löschen: Den einen, der sich in der EXECUTION-Queue befindet. Denn das ist der Job, mit dem Sie zur Zeit eingeloggt sind.***

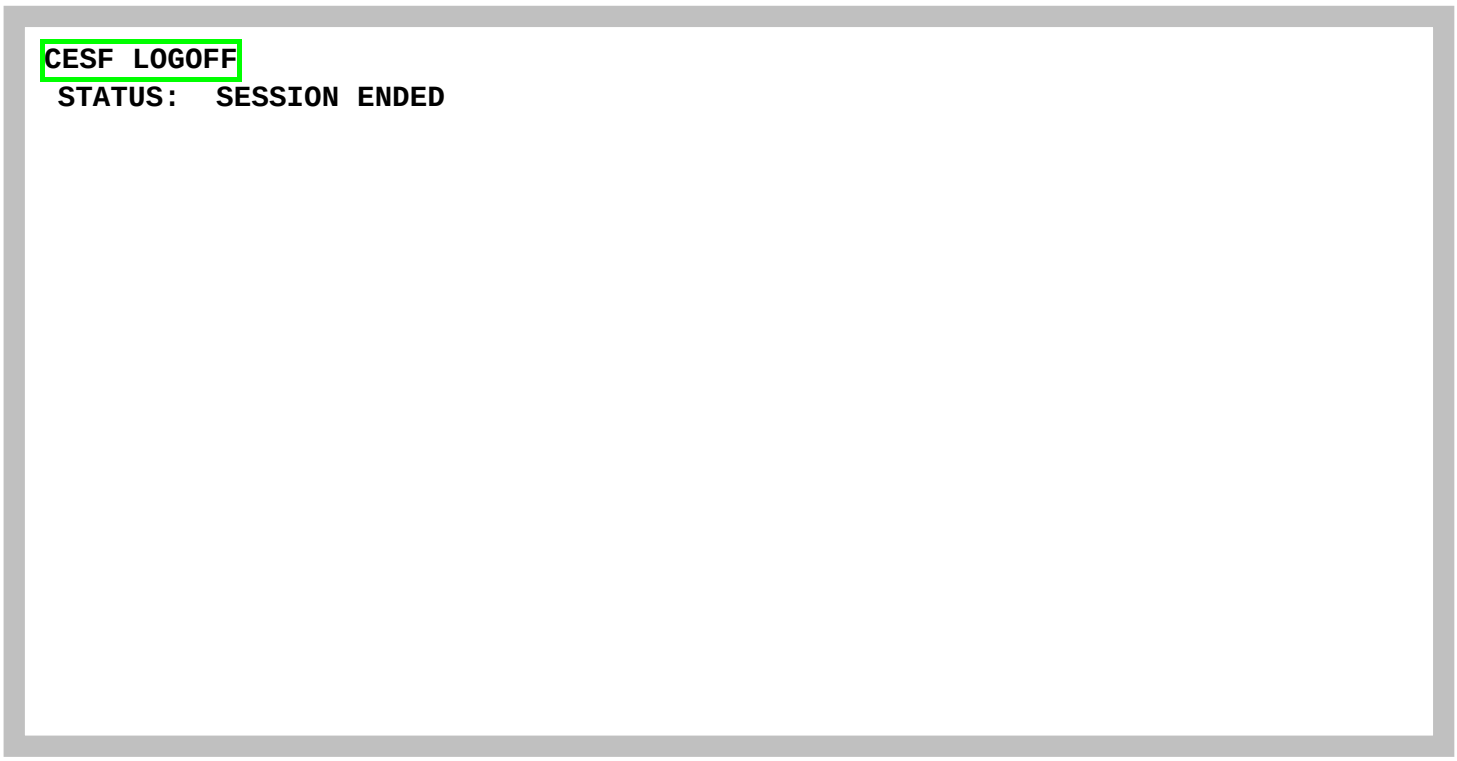

Abbildung 32: Ausloggen aus CICS

Die Ausführung unserer Transaktion (unseres COBOL-Programms) ist damit abgeschlossen – CICS erwartet jetzt die Eingabe einer neuen Transaktion. Dies könnte z.B. CEDA DISPLAY GROUP(*) sein.

Wenn wir mit unserer CICS Sitzung fertig sind und keine weitere Transaktion durch Eingabe einer TRID starten wollen, geben wir die Logoff-Transaktion "CESF LOGOFF" ein, gefolgt von der Eingabetaste, ein (s. Abbildung 32).

7. Anhang

Die Übersetzung des COBSTA05 erzeugt MAXCC=8

Dieses Problem kann aufgetreten sein, weil der in unserem JCL-Script verwendete Zeiger schon existiert.

Sehen sie hierzu im ISPF-Menu More -> SDSF -> Status of jobs nach, welchen Fehler sie erhalten. Sollte es tatsächlich daran liegen, dass der Zeiger schon existiert, so finden sie hier die Meldung:

```
READY
  DSN S(D931)
DSN
  BIND PLAN(ZGR218CO) MEMBER(COB218) ACTION(REP) RETAIN ISOLATION(CS)
DSNT210I  -D931 BIND AUTHORIZATION ERROR
          USING PRAK218 AUTHORITY
          PLAN = ZGR218
          PRIVILEGE = BIND
DSNT201I  -D931  BIND FOR PLAN ZGR218CO   NOT SUCCESSFUL
DSN
END
READY
END
```

In diesem Fall überlegen sie sich bitte einen anderen Namen für ihren Zeiger. Verwenden sie jedoch immer die Ziffern ihres Accounts.

"CEDA INSTALL GROUP ..." erzeugt den Fehler "install unsuccessful"

Dieses Problem könnte auftreten, wenn jemand mehrmals den Befehl "CEDA INSTALL GROUP ..." eingibt. Als Fehlermeldung wird INSTALL UNSUCCESSFUL zurückgegeben.

Das Problem ist, dass sich die schon einmal per "CEDA INSTALL GROUP ..." installierte DB2ENTRY-Komponente nicht so ohne weiteres überschreiben lässt.

Man muss das Überschreiben erlauben. Dies erfordert aber Administrator Rechte über die Ihre User ID nicht verfügt. Deshalb versuchen Sie bitte, das Problem zu vermeiden, indem ein mehrfaches Abarbeiten der Tutorials 5 vermieden wird.

Sollte das Problem trotzdem einmal auftreten, informieren Sie bitte jemanden mit Admin-Rechten, z. B. Ihren Betreuer.

Zu Ihrer Information:

Das Problem kann mit Administrator Rechten behoben werden durch die Eingabe:

CEMT I DB2E(<DB2E-Name>)

Dies gibt den DB2Entry mit dem Namen <DB2E-Name> auf dem Bildschirm aus.

Im konkreten Beispiel liefert

CEMT I DB2E(A060) die folgende Bildschirmausgabe:

```
I DB2E(A060)
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Db2e(A060    ) Txi Sig Ena Poo Hig Pro( 0000 ) Pth(0000)
    Threadl( 0003 ) Threads(0000) Twa Plan( AS5      )
```

Der Wert "Ena" (ENAble) ist auf "Dis" (DISable) zu setzen, um ein Überschreiben zu erlauben. Dazu reicht es, wenn man den Buchstaben "E" von "Ena" mit einem "D" überschreibt sowie die Eingabetaste betätigt. Das Ergebnis dieser Aktion ist im konkreten Beispiel

```
I DB2E(A060)
STATUS:  RESULTS - OVERTYPE TO MODIFY
 Db2e(A060    ) Txi Sig Dis Poo Hig Pro( 0000 ) Pth(0000)
 NORMAL
    Threadl( 0003 ) Threads(0000) Twa Plan( AS5      )
```

Nun ist ein Überschreiben des DB2Entry-Eintrages "A060" wieder möglich und damit auch eine Neuinstallation der Gruppe PRAK218 wieder möglich:

CEDA INSTALL GROUP(PRAK218)

funktioniert fehlerfrei und gibt wieder

INSTALL SUCCESSFUL zurück.